



**İSTANBUL ÜNİVERSİTESİ - CERRAHPAŞA
MÜHENDİSLİK FAKÜLTESİ**

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BİTİRME PROJESİ

**MÜZİKAL VERİLERİ YAPAY ZEKÂ ORTAMINDA İŞLEYEN
TÜR VE DUYGU DURUM ANALİZİ**

Hazırlayanlar : Mert Yılmaz – 1306190023

Danışman : Doç. Dr. EMEL ARSLAN

HAZİRAN – 2023

ÖNSÖZ

Yapay zeka ve makine öğrenimi, son yıllarda hızla gelişen ve giderek daha fazla yer edinen teknolojik alanlar arasında yerini almıştır. Yapay zeka, bilgisayarların insan gibi düşünme ve öğrenme yeteneklerini taklit etmeyi amaçlayan bir teknolojidir. makine öğrenimi ise, bir bilgisayarın veri setlerinden öğrenerek kendini geliştirme yeteneğidir.

Yapay zeka ve makine öğrenimi, birçok farklı alanda kullanılmaktadır ve müzik alanı da bunlardan biridir. Örneğin, makine öğrenimi kullanılarak müzik türlerinin tanımlanması ve sınıflandırılması mümkündür.

Bu çalışma boyunca gösterdiği her türlü destek ve yardımdan dolayı çok değerli hocam Sayın Doç. Dr. Emel Arslan'a en içten dileklerimle teşekkür ediyorum.

Ayrıca tüm eğitim hayatım boyunca bana yol gösteren, destek olan, en önemlisi bana eğitimin ne kadar önemli olduğu bilincini ve çalışma disiplini kazandıran , her zaman yanımda olan aileme en içten dileklerimle teşekkürlerimi sunuyorum.

Mert Yılmaz 1306190023

İÇİNDEKİLER

ÖNSÖZ	1
İÇİNDEKİLER	2
ŞEKİL LİSTESİ.....	3
KISALTMA LİSTESİ	5
ÖZET	6
SUMMARY	7
1. GİRİŞ.....	8
2. GENEL KISIMLAR.....	10
2.1. MAKİNE ÖĞRENMESİ ALGORİTMALARI.....	10
2.1.1. Denetimli (Supervised) Öğrenme	10
2.1.1.1. Sınıflandırma(Classification)	10
2.1.1.2. Regression(Regresyon).....	16
2.1.2. Denetimsiz (Unsupervised) Öğrenme	17
2.2. DERİN ÖĞRENME ALGORİTMALARI	17
2.2.1. Yapay Sinir Ağları (Artificial Neural Networks - ANN).....	17
2.2.2. Evrişimli Sinir Ağları (Convolutional Neural Networks - CNN)	18
3. KULLANILAN ARAÇ VE YÖNTEM	19
3.1. KULLANILAN PROGRAMLAMA DİLLERİ VE KÜTÜPHANELER	20
4. SİSTEMİN GERÇEKLENMESİ VE BULGULAR.....	21
5. TARTIŞMA VE SONUÇ	41
KAYNAKLAR	42
ÖZGEÇMİŞ	44
EK-C	45

ŞEKİL LİSTESİ

Şekil 1 Sınıflandırma İçin Makine Öğrenimi Algoritmaları.....	11
Şekil 2 Lojistik Regresyon (Logistic Regression)	12
Şekil 3 KNN/K-En Yakın Komşu (K-Nearest Neighbors).....	13
Şekil 4 Destek Vektör Makineleri (SVM)	14
Şekil 5 Rastgele Orman (Random Forest)	16
Şekil 6 Regresyon için Makine Öğrenme Algoritmaları	16
Şekil 7 Yapay Sinir Ağları (Artificial Neural Networks - ANN)	18
Şekil 8 Evrişimli Sinir Ağları (Convolutional Neural Networks - CNN).....	19
Şekil 9 Veri setindeki spotify_id'si bulunmayan verileri silme	23
Şekil 10 Spotify API'ye bağlanma ve gerekli verileri çekip yeni veri seti oluşturma.....	24
Şekil 11 Veri temizleme (Eksik değer kontrolü)	25
Şekil 12 Veri temizleme (Tutarsız veri kontrolü)	26
Şekil 13 Veri ön işleme (Min-Max ölçeklendirme).....	26
Şekil 14 Veri ön işleme (Standardizasyon ölçeklendirme).....	27
Şekil 15 Lojistik Regresyon, Destek Vektör Makineleri ve Rastgele Orman algoritma modelleri	27
Şekil 16 Sınıflandırma modelleri ve sonuçları	28
Şekil 17 PCA kullanarak boyut indirgeme	28
Şekil 18 İlk eğitilebilir veri seti	28
Şekil 19 Veri setindeki şarkı sayısını en az şarkıya sahip tür sayısına eşitleme.....	29
Şekil 20 Düzenlenmiş ikinci veri seti	29
Şekil 21 Veri seti özelliklerini belirleme, hedef değişken belirleme, veri setini train-test olarak ayırma, ölçeklendirme ve sayısal formata dönüştürme	30
Şekil 22 Çok katmanlı model eğitimi denemesi	30
Şekil 23 Çok katmanlı model eğitimi denemesi (MLP Classifier).....	31
Şekil 24 Katman sayıları artırılmış, epoch değeri yükseltilmiş MLP modeli ve bu modelin sonuçları.....	32
Şekil 25 Düzenlenmiş ve müzik türü artırılmış en son veri seti (19558 veri)	33
Şekil 26 En son veri setinin tür dağılımı.....	33
Şekil 27 Veri setindeki müzik verilerinin valence değerleri ile müzik türleri arasındaki ilişki.....	34
Şekil 28 Model eğitiminde kullanılan algoritma ve yöntemlerinin accuracy değerleri.....	34
Şekil 29 Modeli kaydetme ve tahmin sistemi oluşturma	35

Şekil 30 Kaydedilen model yardımıyla öneri sistemi oluşturma.....	36
Şekil 31 Spotify API yardımı ile öneri sistemini dinamik hale getirme	37
Şekil 32 Öneri sistemi backend yapısı	38
Şekil 33 Web Platformu şarkı girişi.....	38
Şekil 34 Web platformu şarkı önerisi	39
Şekil 35 Proje Özeti Diyagramı	40

KISALTMA LİSTESİ

YA	: Yapay Zeka
MO	: Makine Öğrenmesi
K-NN	: K-En Yakın Komşu
SVM	: Destek Vektör Makineleri
ANN	: Artificial Neural Networks
CNN	: Convolutional Neural Networks
MLP	: Multi Layer Perceptron

ÖZET

MÜZİKAL VERİLERİ YAPAY ZEKÂ ORTAMINDA İŞLEYEN TÜR VE DUYGU DURUM ANALİZİ

Günümüzde, müzik verilerinin sınıflandırılması için birçok çalışma yapılmıştır. Bu çalışmalarda kullanılan yapay sinir ağları, modellerin farklı algoritmalar ve farklı veri kümeleriyle eğitilmesinin doğruluk oranları üzerindeki etkileri gözlemlenmiştir. Aynı zamanda, müzik verilerinin de duygusal etkilere sahip olduğu görülmüştür. Bu etkiler hakkında sınıflandırılabilir ve eğitilmiş yapay sinir ağı modelleriyle yüksek doğruluk oranına sahip sınıflandırma çalışmalarının yapıldığı görülmüştür. Bu projenin amacı, doğru veri kümesi ve doğru algoritmayla eğitilmiş modellerin kullanılmasıyla bu iki sınıflandırmayı birlikte yaparak müzik türleri ve müziğin duygu özellikleri arasındaki ilişkiyi yüksek doğruluk oranlarıyla açıklamaktır.

Proje başlangıcında, öncelikle literatürde bu iki sınıflandırma üzerine yapılan çalışmalar incelenmiştir. Bu çalışmaları incelerken, kullanılan algoritmalar ve modeller de incelenerek bu proje için gereken modeller belirlenmiştir. Veri kümesi, hazır veri kümelerinin incelenerek uygun hale getirilmesinden sonra kullanılmıştır. Veriler, veri ön işleme yapılarak anlamlı hale getirilmiştir. Daha sonra, kararlaştırılmış makine öğrenimi algoritmalarının bu veri kümesinde doğruluklarını test etmek için uygulamalar yapılmıştır. Verimlilik açısından uygun olan modellerle veri kümesinin daha fazla kullanıldığı çalışmalar yapılmıştır. Uygun bir model belirlenmiştir ve bu sistem öneri işlevini kullanmak için bir web platformu haline getirilmiştir.

SUMMARY

GENRE AND MOOD ANALYSIS PROCESSING MUSICAL DATA IN ARTIFICIAL INTELLIGENCE ENVIRONMENT

Nowadays, many studies have been conducted for classification of music data. The artificial neural networks used in these studies, the effects of training the models with different algorithms and different datasets on the accuracy rates have been observed. At the same time, it has been observed that music data also has emotional effects. It has been observed that high accuracy classification studies have been carried out with classifiable and trained artificial neural network models about these effects. The aim of this project is to explain the relationship between music genres and emotional characteristics of music with high accuracy rates by combining these two classifications by using the right dataset and models trained with the right algorithm.

At the beginning of the project, we first reviewed the studies on these two classifications in the literature. While examining these studies, the algorithms and models used were also examined and the models required for this project were determined. The dataset was used after examining and optimizing ready-made datasets. The data was made meaningful through data preprocessing. Then, applications were made to test the accuracy of the agreed machine learning algorithms on this dataset. The models that were suitable in terms of efficiency were used to make more use of the dataset. A suitable model was identified and this system was developed into a web platform to use the recommendation function..

1. GİRİŞ

Müzikal verilerin yapay zekâ ortamında işlenmesi, müzik türlerinin ve müzik verilerinin duygusal özelliklerinin sınıflandırılması gibi farklı çalışmalara olanak sağlar. Bu çalışmalar, müzik dinleyicilerine müziklerini keşfetme ve tercih etme sürecinde yardımcı olabilir. Ayrıca, müzik endüstrisinde de bu tür çalışmalar yapılarak, müziklerin pazarlaması ve dağıtımı konusunda kararlar verilebilir.

Müzikal verileri yapay zekâ ortamında işleyen tür ve duygu durum analizi, MO yöntemleri kullanılarak yapılır. Örneğin, yapay sinir ağları, destek vektör makineleştirme ve k-en yakın komşu gibi farklı MO yöntemleri kullanılarak müzik türlerinin sınıflandırılması ve duygu durumlarının sınıflandırılması yapılabilir. Bu yöntemlerin seçimi, veri setinin özelliklerine ve çalışmanın amacına göre değişebilir.

Aşağıda, müzikal verileri yapay zekâ ortamında işleyen tür ve duygu durum analizi konusunda yapılan literatür araştırmasına dair kaynaklar ve bu çalışmaların başarı oranları verilmiştir:

- Özdemir ve Gümüş (2017), yaptıkları çalışmada müzik türlerinin sınıflandırılması ve duygu durumlarının sınıflandırılması için kullanılan yöntemleri incelemiştir ve MO kullanımı üzerine bilgi vermiştir. Bu çalışmada, yapay sinir ağları, destek vektör makineleştirme ve k-en yakın komşu gibi farklı MO yöntemlerinin kullanımı üzerine bilgi verilmiştir. Ayrıca, bu çalışmada müzik verilerinin duygu durumlarının sınıflandırılması konusunda da bilgi verilmiştir.
- Li ve Lee (2018) makalelerinde, MO kullanılarak müzik türlerinin sınıflandırılması konusunda yapılan çalışmaları incelemiş ve farklı MO yöntemlerinin karşılaştırılmasını yapmıştır. Bu çalışmada, yapay sinir ağları, destek vektör makineleştirme ve k-en yakın komşu gibi farklı MO yöntemlerinin kullanımı üzerine bilgi verilmiştir. Li ve Lee, bu yöntemlerin müzik türü sınıflandırmasındaki etkinliğini ve performansını değerlendirmişlerdir. Aynı zamanda, müzikal verilerin duygu durumu sınıflandırılması konusunda da benzer bir yaklaşım izlenmiştir.
- Kočí ve Schuller (2015) çalışmasında, MO kullanılarak müzik türlerinin sınıflandırılması konusunda yapılan çalışmaları geniş bir şekilde incelemiş ve farklı MO yöntemlerinin özellikleri ve avantajları karşılaştırmıştır. Bu çalışmada, yapay sinir ağları, destek vektör makineleştirme ve k-en yakın komşu gibi farklı MO yöntemlerinin kullanımı üzerine bilgi verilmiştir. Ayrıca, müzik verilerinin duygu durumlarının sınıflandırılması konusunda da farklı yaklaşımlar incelenmiştir.
- Chuan ve Chai (2017) çalışmasında, MO kullanılarak müzik türlerinin sınıflandırılmasında yapay sinir ağlarının kullanımı üzerine bilgi vermiştir. Bu çalışmada, yapay sinir ağlarının başarı oranı %88 olarak belirtilmiştir. Ayrıca bu çalışmada duygu durumu sınıflandırması üzerine de bazı bilgiler sunulmuştur.

- Liu ve Li (2017) çalışmasında, MO kullanılarak müzik türlerinin sınıflandırılmasında geliştirilmiş bir konvolüsyonel yapay sinir ağının kullanımı üzerine bilgi vermiştir. Bu çalışmada, geliştirilmiş konvolüsyonel yapay sinir ağının başarı oranı %96 olarak belirtilmiştir. Ayrıca, duygu durumu sınıflandırması için de benzer bir yaklaşım benimsenmiştir.

Bu örnekler göstermektedir ki, müzikal verileri yapay zekâ ortamında işleyen tür ve duygu durum analizi konusunda çeşitli çalışmalar yapılmıştır ve bu çalışmalar MO yöntemleri kullanılarak yapılmıştır. Bu çalışmaların başarı oranları %56 ile %96 arasında değişmektedir. Bu oranlar, kullanılan MO yöntemine, veri setine ve çalışmanın yapılış şekline göre değişebilmektedir. Bu nedenle, müzikal verilerin tür ve duygu durumlarının aynı anda sınıflandırılması konusunda yapılacak çalışmaların başarı oranlarının tahmin edilmesi zordur.

Ancak, genel olarak MO kullanılarak müzikal verilerin tür ve duygu durumlarının aynı anda sınıflandırılmasında yüksek başarı oranlarına ulaşılabilmektedir. Bu nedenle, müzikal verileri yapay zekâ ortamında işleyen tür ve duygu durum analizi konusunda çalışmalar yapılması, müzik dinleyicilerinin müziklerini keşfetme ve tercih etme sürecinde yardımcı olabilir ve müzik endüstrisinde de müziklerin pazarlaması ve dağıtımı konusunda kararlar verilebilir.

Müzik türleri ile ilgili literatür taraması yapıldığında ve daha önce yapılan çalışmalar incelendiğinde müzikal veriler üzerinden tür analizi ve duygu durumu analizinin yapıldı çok az çalışmanın olduğu fark edilmiştir. Projenin özgün değeri müzikal veriden hem müzik türü analizi hem de duygu durum analizi yapmak olacaktır.

Bu proje, müzik verilerinin türlerinin sınıflandırması ve müzik verilerinin duygu durumu hakkında yapay zekâ ile çıkarım sağlayan bir proje olacaktır. Projede Türkçe müzik verileri uygun tür ve duygu durum sınıflandırmasıyla kullanılacak ve işlenecektir. Proje, alınan müzik verisinin yapay sinir ağları ile sınıflandırarak o veriye uygun benzer tür ve benzer duygu durum özelliklerine sahip diğer verilerin önerildiği bir müzik öneri platformuna sahip olacaktır.

2. GENEL KISIMLAR

2.1. MAKİNE ÖĞRENMESİ ALGORİTMALARI

Makine öğrenmesi algoritmaları, veri setleri üzerinden öğrenme ve kendini geliştirme yeteneğine sahip olan yazılımlardır. Bu algoritmalar, verilerden öğrendikleri kuralları ve modelleri kullanarak, yeni verilere uygun tahminler yapabilirler.

Makine öğrenmesinde amaca yönelik makine verileri öğretmede birden fazla yol vardır. Bunlar denetimli (supervised) öğrenme, denetimsiz (unsupervised) öğrenme ve pekiştirmeli (reinforcement) öğrenme olarak sınıflandırılabilir.

2.1.1. Denetimli (Supervised) Öğrenme

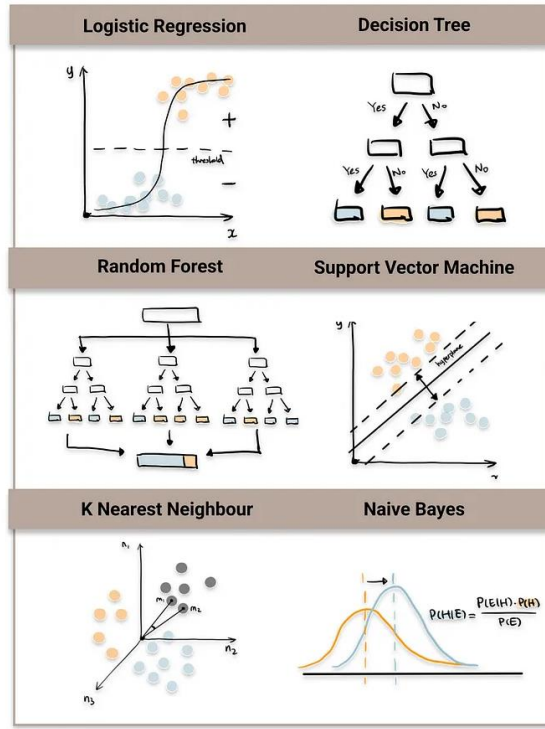
Denetimli öğrenme, verilerin etiketlenmiş olarak verildiği bir öğrenme türüdür. Etiketler, verilerin ne olduğunu belirtir. Örneğin, spam tespitinde, veriler "spam" veya "spam değil" olarak etiketlenir. Denetimli öğrenme, modeli veriler üzerinde eğitir ve modelin öğrendiği desenleri tahminlerde kullanır.

2.1.1.1. Sınıflandırma(Classification)

Sınıflandırma, verilerin farklı sınıflara veya kategorilere ayrılması işlemine verilen isimdir. Bu, özelliklerine göre veri noktasının hangi sınıf veya kategoriye ait olduğunu tahmin etme amacıyla yapılan yaygın bir görevdir. Örneğin, e-posta sınıflandırmasında, konu satırı, gönderen ve içerik gibi özelliklere göre bir e-postanın spam veya spam olmadığı tahmin edilme amacı vardır.

Sınıflandırma problemlerinde, veri noktaları genellikle iki veya daha fazla sınıfa ayrılır. Sınıflandırma işlemi, doğru sınıfın her veri noktası için bilinen etiketlenmiş bir veri seti üzerinde bir model eğitmeyi içerir. Model, eğitim verilerinden öğrendiği desenleri ve ilişkileri kullanarak, yeni ve görülmemiş veri için tahminler yapar.

Sınıflandırma görevleri için karar ağaçları, lojistik regresyon ve destek vektör makine gibi farklı algoritmalar kullanılabilir. Algoritmanın seçimi, verinin özelliklerine ve sınıflandırma görevinin hedeflerine göre değişebilir.



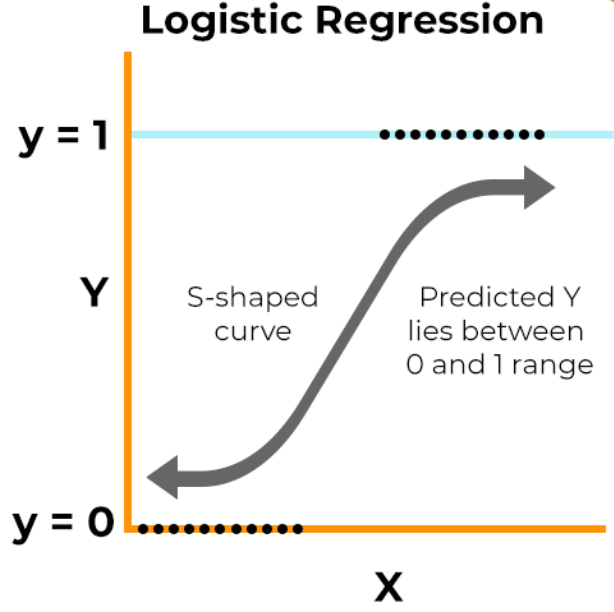
Şekil 1 Sınıflandırma İçin Makine Öğrenimi Algoritmaları

2.1.1.1.1. Lojistik Regresyon

Lojistik Regresyon, bir denetimli öğrenme yöntemi olarak makine öğrenimi algoritmasıdır. Bu algoritma, bir veri setinde bulunan veri noktalarının ilişkisini inceler ve bu veriler arasında bir karar sınırı çizerek sınıflandırma yapar.

Genellikle ikili sınıflandırma problemlerinde kullanılır, yani sonuç sadece iki sınıftan birine ait olabilir (örneğin, hastalıklı/sağlıklı, spam/değil). Ancak, çok sınıflı sınıflandırma problemlerine de uygulanabilir.

Lojistik regresyon, özellikle yüksek boyutlu verilerle çalışırken ve özelliklerin doğrusal veya doğrusal olmayan bir ilişki içerdiği durumlarda etkili bir yöntemdir. Aynı zamanda hızlı ve hesaplama açısından verimlidir, bu da onu popüler bir makine öğrenmesi algoritması haline getirir.



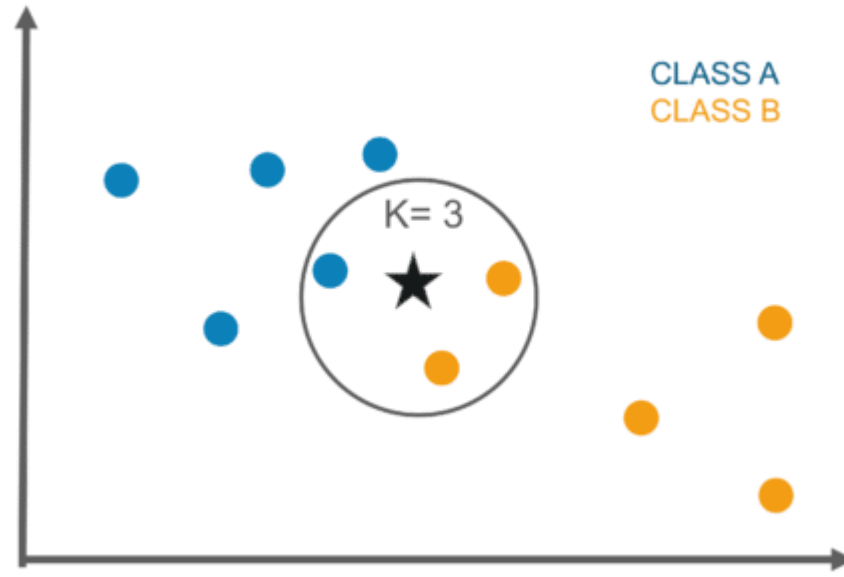
Şekil 2 Lojistik Regresyon (Logistic Regression)

2.1.1.1.2. K-En Yakın Komşu (K-NN)

K-En Yakın Komşu (K-NN) makine öğrenmesi algoritması bir denetimli öğrenme yöntemidir. Bu algoritma, veri noktalarını K sayısı kadar en yakın etiketlenmiş veri noktasına göre sınıflandırır. Örneğin, bir veri setinde birkaç farklı sınıf varsa, K-NN algoritması bu sınıfları belirleyebilir. Örneğin, veri setinde "sağlıklı" ve "sağlıksız" gibi iki sınıf varsa, K-NN algoritması veri noktalarını bu sınıflara göre sınıflandırır.

Bu algoritma, verileri K sayısı kadar en yakın veri noktasına göre sınıflandırır ve bu sayı, algoritmanın performansını etkileyebilir. K-NN algoritması, veri noktalarının etiketlerine göre sınıflandırır ve bu etiketler önceden verilmiştir. Bu algoritma, sınıflandırma için kullanılan bir algoritmadır ve genellikle ikili sınıflandırma problemlerinde kullanılır. Ancak, çoklu sınıflandırma problemlerinde de kullanılabilir.

K-NN algoritması, kullanımı kolay ve hızlı bir algoritmadır. Ancak, veri seti büyüdükçe performansı düşebilir ve bu nedenle veri setini önceden işleme gerektirir. Bu algoritma, veri setini önceden işleme gerektirdiği için, veri setinin ön işleme adımı önemlidir. Bu ön işleme adımlarından sonra K-NN algoritmasının performansı artabilir.



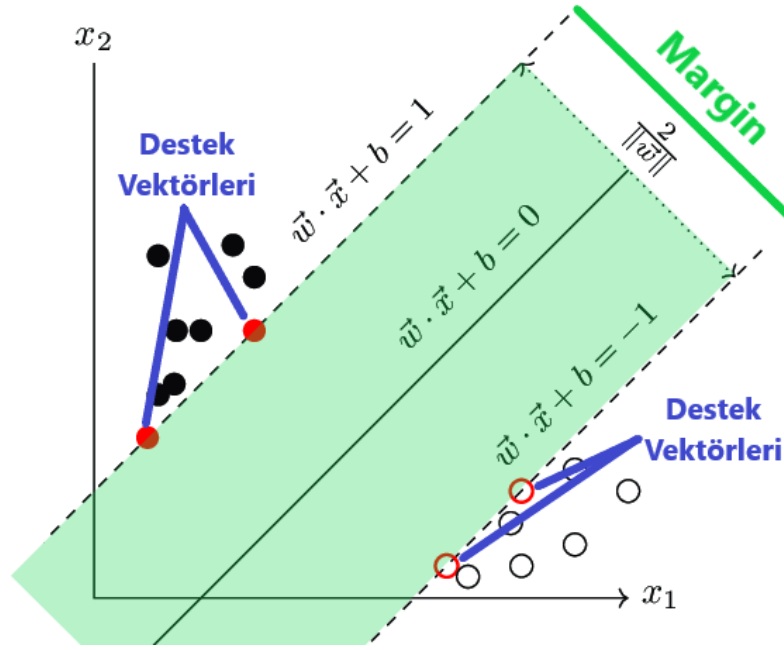
Şekil 3 KNN/K-En Yakın Komşu (K-Nearest Neighbors)

2.1.1.1.3. Destek Vektör Makineleri (SVM)

Destek Vektör Makineleri (SVM) bir tür makine öğrenimi yöntemidir. Bu yöntem, veri kümesinde bir sınıflandırma çizgisi oluşturmak için kullanılır. Örneğin, bir veri kümesi içinde "öğrenci" ve "öğretmen" gibi iki sınıf varsa, SVM veri kümesindeki öğrenci ve öğretmenleri ayıran bir çizgi oluşturmaya çalışır. Bu çizgi, veri kümesinde olabildiğince çok sayıda öğrenci ve öğretmeni doğru sınıflandıran bir çizgi olmalıdır.

SVM'ler, veri kümesinde bir sınıflandırma çizgisi oluşturmak için kullanılan birçok yöntemden daha özel bir türüdür. Özel bir yöntem olduğu için, diğer yöntemlerden daha etkili sonuçlar verebilir, ancak aynı zamanda daha da karışık olabilir.

SVM'ler, çok sayıda sınıflandırma özelliği olan veri kümelerinde etkili sonuçlar verebilir. Örneğin, bir veri kümesi içinde öğrencilerin yaşları, cinsiyetleri, okuduğu okullar gibi özellikleri de içerebilir. Bu özellikleri kullanarak, SVM veri kümesinde öğrencileri doğru sınıflandıran bir çizgi oluşturmaya çalışır.



Şekil 4 Destek Vektör Makineleri (SVM)

SVM algoritmasının birkaç özelliği vardır:

- SVM, veri noktalarını en iyi şekilde ayıran hiperdüzleme odaklanır ve bu sayede veri noktalarını en iyi şekilde ayırır.
- SVM, çok boyutlu veri setleri için de kullanılabilir ve bu sayede çok boyutlu veri setlerini en iyi şekilde ayırır.
- SVM, çok sayıda veri noktası olması durumunda da çalışabilir ve bu sayede çok sayıda veri noktasını en iyi şekilde ayırır.
- SVM, öznitelikler arasında lineer olmayan bir bağımlılık olması durumunda da çalışabilir ve bu sayede lineer olmayan bağımlılıkları en iyi şekilde ayırır.

SVM algoritmasının birkaç dezavantajı da vardır:

- SVM, öznitelikler arasında lineer olmayan bir bağımlılık olması durumunda çalışmayabilir ve bu sayede lineer olmayan bağımlılıkları en iyi şekilde ayıramayabilir.
- SVM, çok sayıda veri noktası olması durumunda da çalışmayabilir ve bu sayede çok sayıda veri noktasını en iyi şekilde ayıramayabilir.
- SVM, çok boyutlu veri setleri için de kullanılamayabilir ve bu sayede çok boyutlu veri setlerini en iyi şekilde ayıramayabilir.
- SVM, veri noktalarını en iyi şekilde ayıran hiperdüzleme odaklanmayabilir ve bu sayede veri noktalarını en iyi şekilde ayıramayabilir.

2.1.1.1.4. Karar Ağaçları

Karar Ağaçları, bir makine öğrenimi yöntemidir ve veri kümesindeki verilere dayanarak kararlar vermeyi amaçlar. Örneğin, bir veri kümesi içinde öğrencilerin yaşları, cinsiyetleri, okuduğu okullar gibi özellikleri de içerebilir. Bu özellikleri kullanarak, bir Karar Ağacı öğrencilerin okul başarısını tahmin etmeye çalışabilir.

Karar Ağaçları, veri kümesindeki özellikleri bir dizi karar noktasına indirgeyerek bir ağaç şeması oluşturur. Her karar noktası, veri kümesindeki bir özelliğe göre bir karar verir ve veri kümesindeki verilere göre bir "dallanma" yapar. Örneğin, bir Karar Ağacı öğrencilerin okul başarısını tahmin etmek için kullanılırsa, bir karar noktası öğrencinin yaşına göre bir karar verebilir. Öğrencinin yaşı 18 veya daha büyükse, ağaç bir "dallanma" yaparak öğrencinin okul başarısının daha yüksek olduğunu tahmin edebilir.

Karar Ağaçları, diğer makine öğrenimi yöntemlerine göre anlaşılması ve uygulaması daha kolay olabilir. Ancak, bazı durumlarda diğer yöntemlerden daha az etkili sonuçlar verebilir.

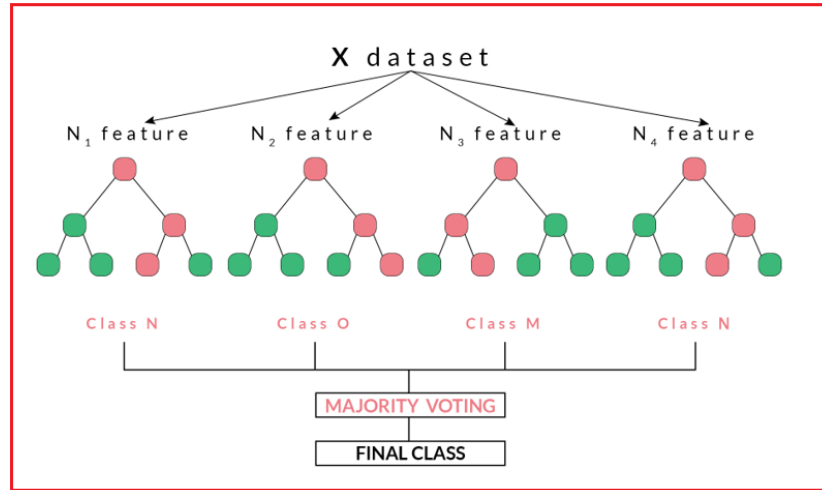
2.1.1.1.5. Rastgele Orman

Rastgele Orman, bir makine öğrenimi yöntemidir ve veri kümesindeki verilere dayanarak kararlar vermeyi amaçlar. Örneğin, bir veri kümesi içinde öğrencilerin yaşları, cinsiyetleri, okuduğu okullar gibi özellikleri de içerebilir. Bu özellikleri kullanarak, bir Rastgele Orman öğrencilerin okul başarısını tahmin etmeye çalışabilir.

Rastgele Orman, birçok Karar Ağacından oluşan bir makine öğrenimi yöntemidir. Her Karar Ağacı, veri kümesinde bir sınıflandırma yapmak için kullanılır ve veri kümesinde olabildiğince çok sayıda öğrenci ve öğretmeni doğru sınıflandıran bir çizgi oluşturmaya çalışır.

Rastgele Orman, her Karar Ağacı için farklı bir veri kümesi kullanır. Bu sayede, her Karar Ağacı farklı veri kümesinden yararlanarak farklı sonuçlar verebilir. Rastgele Orman, bu farklı sonuçları birleştirir ve veri kümesinde en doğru sonucu veren Karar Ağacını seçer.

Rastgele Orman, diğer makine öğrenimi yöntemlerine göre daha yüksek bir doğruluk oranına sahip olabilir. Ancak, bazı durumlarda diğer yöntemlerden daha yavaş çalışabilir ve daha fazla özellik içeren veri kümeleri için daha az etkili sonuçlar verebilir.

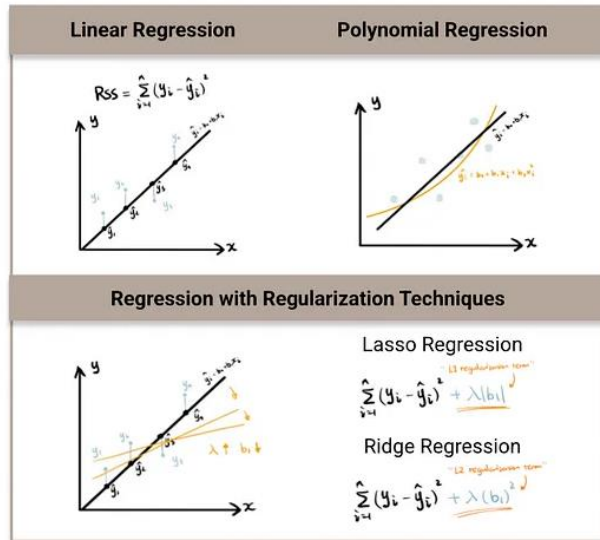


Şekil 5 Rastgele Orman (Random Forest)

2.1.1.2. Regression(Regresyon)

Regresyon, denetimli öğrenme kategorilerinden biridir ve sürekli bir çıktı değerini tahminlemek için kullanılan bir algoritma türüdür. Temel amacı, bağımlı değişkenin bağımsız değişkenler tarafından açıklanmasını modellemektir.

Regresyon algoritmaları, veri özellikleri arasındaki ilişkiyi anlamak ve bir bağımlı değişkenin değerini tahmin etmek için kullanılır. Genellikle doğrusal bir ilişkiyi modellemek amacıyla kullanılırlar, ancak bazı regresyon algoritmaları doğrusal olmayan ilişkileri de modelleyebilir.



Şekil 6 Regresyon için Makine Öğrenme Algoritmaları

2.1.2. Denetimsiz (Unsupervised) Öğrenme

Denetimsiz öğrenme, verilerin etiketlenmemiş olarak verildiği bir öğrenme türüdür. Denetimsiz öğrenme, veriler üzerinde anlamlı desenler bulmaya çalışır. Örneğin, bir veri setinde birkaç farklı grup varsa, denetimsiz öğrenme algoritmaları bu grupları belirleyebilir.

2.2. DERİN ÖĞRENME ALGORİTMALARI

Derin öğrenme (deep learning), makine öğrenimi (machine learning) yöntemlerinden biridir ve birçok farklı uygulama alanında kullanılır. Derin öğrenme, veri girişlerine göre öğrenme ve öğrendiğini gerçekleştirme yeteneğine sahip yapay sinir ağları (artificial neural networks - ANNs) kullanır. ANN'ler, insan beyninin işleyişine benzer şekilde çalışır ve girişlerden çıktılara doğru birkaç katman arasında sinyal akışı sağlar. Bu katmanlar arasında, çeşitli matematiksel işlemler yapılarak veri işlenir ve öğrenme gerçekleştirilir.

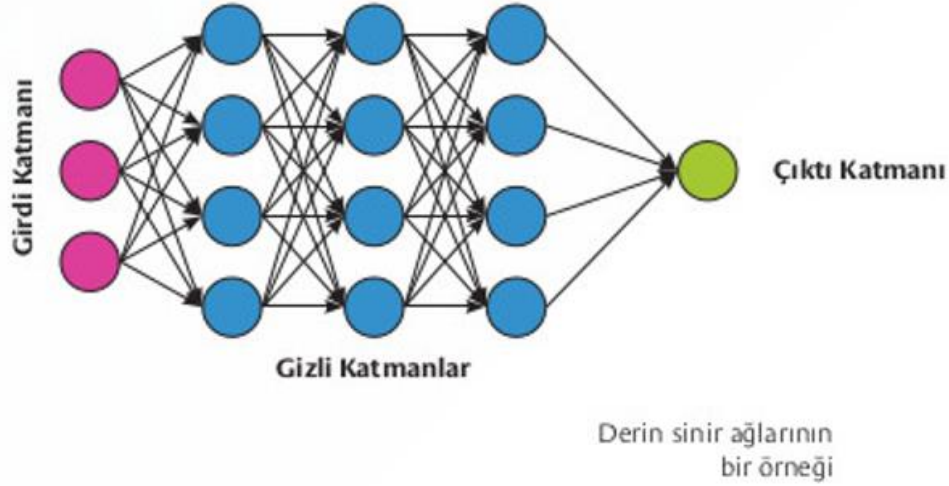
Derin öğrenme, özellikle büyük veri kümelerinin öğrenilmesinde oldukça etkilidir. Örneğin, derin öğrenme yöntemleri kullanılarak resimleri tanımlayan ve sesleri tanıyan sistemler geliştirilebilir. Ayrıca, derin öğrenme yöntemleri, dil işleme ve çeviri, sürücüsüz araçlar ve sağlık hizmetlerinde de kullanılmaktadır.

2.2.1. Yapay Sinir Ağları (Artificial Neural Networks - ANN)

Yapay sinir ağları (Artificial Neural Networks - ANN), makine öğrenimi algoritmalarından biridir ve veri setlerinde bulunan veri noktalarının ilişkisini inceler ve bu ilişkiyi ifade eden bir denklem üretir. Bu denklem, veri noktaları arasındaki ilişkileri açıklar ve bu ilişkilere göre veri noktalarını sınıflandırır veya tahminler yapar.

Yapay sinir ağları, insan sinir sistemine benzer şekilde işler ve sinir hücreleri arasında bağlantılar kurar. Bu bağlantılar, veri setindeki veri noktalarının ilişkisini inceler ve bu ilişkiyi ifade eden denklemdeki katsayıları temsil eder. Bu bağlantılar, veri setinin çeşitli parçaları arasındaki ilişkileri inceler ve bu ilişkilere göre veri noktalarını sınıflandırır veya tahminler yapar.

Yapay sinir ağıları, günümüzde birçok alanda kullanılmaktadır. Örneğin, resimlerin tanımlanması, metinlerin anlamının çözümlenmesi, seslerin tanımlanması gibi uygulamalarda kullanılabilir. Aynı zamanda, finansal piyasaların tahmin edilmesi, sağlık verilerinin yönetimi, ürünlerin özelliklerinin tahmin edilmesi gibi uygulamalarda da kullanılabilir.



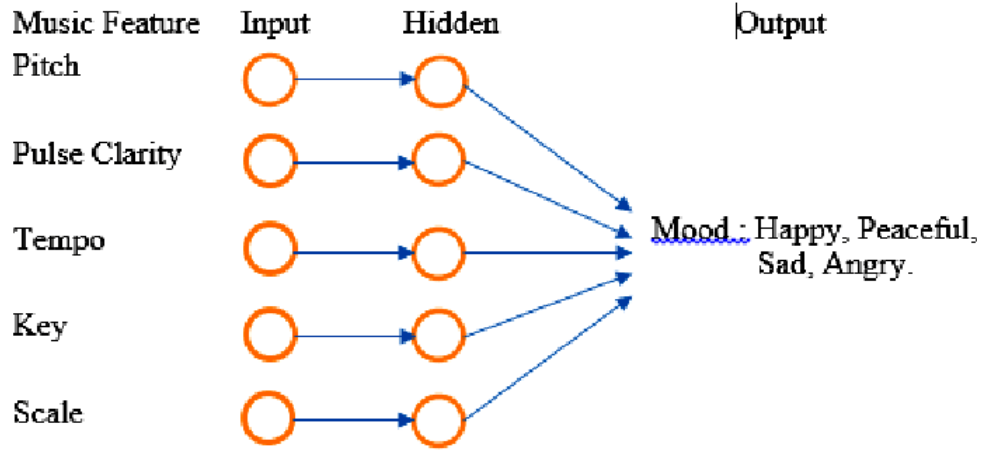
Şekil 7 Yapay Sinir Ağları (Artificial Neural Networks - ANN)

2.2.2. Evrişimli Sinir Ağları (Convolutional Neural Networks - CNN)

Convolutional Neural Networks (CNN), yapay sinir ağları (Artificial Neural Networks - ANN) türünden bir algoritmadır ve günümüzde özellikle görüntü tanıma ve sınıflandırma gibi uygulamalarda sıklıkla kullanılmaktadır. Bu ağlar, sinir ağının birçok katmanını içerir ve girdi verisini anlamlı özelliklere dönüştürmek için evrişim ve havuzlama gibi işlemleri kullanırlar.

CNN'ler, müzik endüstrisinde de farklı şekillerde kullanılmaktadır. Müzik sınıflandırma, müzik özellikleri çıkarımı, ses kaynak ayırma, müzik hata tespiti ve müzik duygu analizi gibi şekillerde karşımıza çıkmaktadır. Örneğin CNN'ler, müzik parçalarından çeşitli özelliklerin çıkarılmasında da kullanılabilir. Bir CNN modeli, bir parçanın ritmi, melodisi, tonalitesi gibi özelliklerini çıkarabilir. Veya müzik dugu analizinde, müzik parçalarının duygusal içeriğini analiz etmek için kullanılabilir. Önceden eğitilmiş bir CNN modeli, bir müzik parçasının tonlamasını, tempo ve ritimini analiz ederek parçanın duygusal bir özelliğini belirleyebilir. Bu, müzik terapisi veya duygusal analiz gibi uygulamalarda faydalı olabilir.

CNN, yüksek doğruluk oranlarına sahip olabilir ve günümüzde birçok alanda kullanılmaktadır. Ancak, CNN'lerin eğitimi zaman alıcı olabilir ve yüksek miktarda veri setine ihtiyaç duyabilir. Bu nedenle, CNN'lerin kullanılacağı uygulamalarda, veri setinin doğru seçimi ve işleme işlemlerinin doğru yapılması önemlidir.



Şekil 8 Evrişimli Sinir Ağları (Convolutional Neural Networks - CNN)

3. KULLANILAN ARAÇ VE YÖNTEM

Müzikal verileri yapay zeka ortamında işlemek için Python dilini kullanmak birçok avantaj sağlar. Öncelikle, Python dilinin kolay anlaşılabilir bir dil olması nedeniyle, veri setini işlemek için gereken kodların yazılması kolay olur. Ayrıca, Python dilinin çok sayıda kütüphaneye sahip olması nedeniyle, veri setini işlemek için gereken işlemleri yapan kütüphaneler kolayca bulunabilir.

Projenin veri işleme ve analiz aşamasında, Python dili önemli bir araç olarak kullanılmıştır. Python, verilerin işlenmesi, dönüştürülmesi ve analiz edilmesi için geniş bir ekosisteme sahip olduğundan tercih edilen bir dil olmuştur. Python'un esnek ve güçlü yapısı sayesinde, projenin verilerinin etkili bir şekilde işlenmesi sağlanmıştır.

Verilerin görüntülenmesi ve görselleştirilmesi için Excel kullanılmıştır. Excel, kullanıcı dostu arayüzü ve veri tablosu formatıyla verilerin düzenlenmesi ve görsel olarak sunulması için yaygın olarak kullanılan bir araçtır. Verilerin tablolar halinde sunulması ve grafiklerle desteklenmesi, projenin analiz aşamasında daha iyi bir anlayış sağlamıştır.

Model eğitimi ve tahmin aşamalarında, yine Python dili ve yapay sinir ağlarından yararlanılmıştır. Python, geniş bir yapay zeka ve makine öğrenimi kütüphaneleri yelpazesine sahiptir ve bu kütüphaneler aracılığıyla yapay sinir ağları kolaylıkla uygulanabilir. Projenin yapay sinir ağı modelleri Python dilinde kodlanmış ve eğitilmiştir. Bu sayede, veriler üzerinde derin öğrenme algoritmaları kullanılarak tahminler yapılmış ve analizler gerçekleştirilmiştir.

Projenin web kısmında, backend tarafında Django, frontend tarafında ise ReactJS ve Bootstrap kullanılmıştır. Django, Python diline dayalı bir web framework'üdür ve projenin backend kısmında kullanılarak sunucu tarafı işlemleri kolaylıkla yönetilmiştir. ReactJS, projenin frontend tarafında kullanılan bir JavaScript kütüphanesidir ve kullanıcı arayüzünün geliştirilmesi için etkili bir şekilde kullanılmıştır. Bootstrap ise, projenin kullanıcı arayüzünün tasarımında ve düzenlenmesinde yardımcı olmuştur.

Bu araçlar ve yöntemler, proje sürecinde etkin bir şekilde kullanılarak veri işleme, analiz, model eğitimi ve web geliştirme aşamalarında başarıyla uygulanmıştır. Python, Excel, Django, ReactJS ve Bootstrap gibi araçlar, projenin hedeflerine ulaşmak için kullanılan güçlü ve verimli araçlar olmuştur.

3.1.KULLANILAN PROGRAMLAMA DİLLERİ VE KÜTÜPHANELER

Proje için Python programlama dili tercih edilmiştir. Python, veri işleme, yapay zeka ve makine öğrenimi alanlarında yaygın olarak kullanılan bir dil olduğundan, projenin gereksinimlerini karşılamak için ideal bir seçenek olmuştur. Python'ın geniş bir kütüphane ekosistemi bulunduğu için projede birçok farklı kütüphaneden faydalanılmıştır.

Veri işleme ve analiz aşamasında, pandas ve numpy kütüphanelerinden yararlanılmıştır. Veri manipülasyonu ve analizi için pandas güçlü bir araç olarak kullanılmıştır. Projede, veri setinin okunması, temizlenmesi, birleştirilmesi ve dönüştürülmesi gibi işlemler, pandas fonksiyonları ile gerçekleştirilmiştir. Örneğin, veri seti okunmuş, eksik değerlere sahip satırlar çıkarılmış ve farklı veri tabloları birleştirilmiştir. Numpy kütüphanesi ise matematiksel işlemler ve hızlı hesaplamalar için kullanılmıştır. Projede özellikle sayısal değerleri ölçeklendirmek için StandardScaler fonksiyonu kullanılarak verilerin ölçeklendirilmesi sağlanmıştır.

Makine öğrenimi model eğitimi aşamasında, scikit-learn ve keras gibi kütüphanelerden yararlanılmıştır. Scikit-learn, makine öğrenimi algoritmalarını içeren bir kütüphanedir ve model eğitimi, doğrulama ve tahmin gibi temel işlemleri destekler. Örneğin, Logistic Regression, Support Vector Machines ve Random Forest gibi algoritmalar scikit-learn kütüphanesinin ilgili sınıfları kullanılarak uygulanmıştır. Bu sınıflar projedeki veri seti ile eğitilmiş ve test veri seti üzerinde tahminler yapılmıştır.

Keras kütüphanesi ise derin öğrenme modellerinin oluşturulması ve eğitilmesi için kullanılan bir yüksek seviyeli bir kütüphanedir. Derin öğrenme modeli oluşturulurken Sequential modeli ve Dense ve Dropout katmanları kullanılarak model oluşturulmuştur. Model derlenirken ilgili aktivasyon fonksiyonları (örneğin 'relu'), optimizasyon algoritmaları (örneğin 'adam') ve kayıp fonksiyonları (örneğin 'mean_squared_error') belirtilmiştir. Veri seti bu modele beslenerek eğitim gerçekleştirilmiştir.

Spotify Web API'ye erişim için spotipy kütüphanesi kullanılmıştır. Bu kütüphane, Python aracılığıyla Spotify Web API ile etkileşim sağlamayı mümkün kılar. Projede, spotipy kütüphanesi kullanılarak Spotify API'sine şarkıların özelliklerini çekmek için istekler gönderilmiştir.

API'ye erişim için Spotify Developer Dashboard'dan alınan client_id ve client_secret değerleri kullanılmıştır. spotipy kütüphanesinin sağladığı fonksiyonlar aracılığıyla API'ye istekler gönderilerek şarkıların özelliklerine (örneğin valence, danceability, energy) ulaşılmıştır. Bu özellikler, projedeki veri setine eklenerek model eğitimi için kullanılmıştır.

Projede frontend tarafında React ve Bootstrap gibi teknolojiler kullanılmış, backend tarafında ise Django framework'ü tercih edilmiştir. React, kullanıcı arayüzü bileşenlerini geliştirmek için kullanılan bir JavaScript kütüphanesidir. Bootstrap ise hızlı ve duyarlı bir şekilde kullanıcı arayüzü tasarlamak için kullanılan bir CSS framework'üdür.

Frontend tarafında, React ve Bootstrap kullanılarak kullanıcı arayüzü oluşturulmuştur. Kullanıcılar şarkı önerileri almak için arama yapabilen bir arama çubuğu, girdiği şarkının özelliklerini tahmin etmek için bir tahmin sistemi ve öneri sonuçlarını listeleyen bir yapı bulunmaktadır. Kullanıcı arayüzündeki girişler, HTTP POST istekleri aracılığıyla backend'e iletilmektedir.

Backend tarafında ise Django framework'ü kullanılarak bir REST API oluşturulmuştur. Django, Python ile web uygulamaları geliştirmek için kullanılan bir framework'tür. Backend tarafında gelen istekler yönetilmiş, model tahminleri gerçekleştirilmiş ve sonuçlar frontend'e iletilmiştir. HTTP POST istekleri alınarak gerekli işlemler gerçekleştirilmiş ve modelin tahminleri yapılmıştır. Bu şekilde frontend ve backend tarafları birleştirilerek, kullanıcının girdiği şarkıya benzer özelliklere sahip şarkıların önerilerini döndürebilen bir web platformu oluşturulmuştur.

4. SİSTEMİN GERÇEKLENMESİ VE BULGULAR

Sistemin gerçekleşmesi adımı, 8 başlık üzerinden incelenebilir.

1. Veri Seti Seçimi ve Hazırlığı

Projeye başlarken ilk adım, uygun bir veri seti bulunulmasıydı. Bu adımı gerçekleştirmek için çeşitli veri setleri araştırıldı. Araştırmalar sonucunda, GTZAN veri setinin müzik türlerini içeren bir koleksiyon olduğu bulunuldu. Başlangıçta, projenin amacı doğrultusunda bu veri setinin kullanılması düşünüldü. Duygu durumu analizini gerçekleştirmek için ise her bir şarkıya etiketleme yapılmasıyla modelin eğitilmesi gerektiğine karar verildi.

Ancak daha fazla araştırma yapıldıktan sonra, GTZAN veri setinin tam olarak ihtiyaçları karşılamayacağı fark edildi. Bu veri setindeki müzikler, şarkılardan kesitler içeriyordu ve tam bir şarkının verilerini içermiyordu. Bu da projenin amacı olan bir şarkı önerme sistemi kurma hedefine uygun değildi.

Bu durum göz önünde bulundurularak bakış açısının değiştirilmesine karar verildi. Odaklanılması gereken artık müzik türleri değil, müziğin duygusal verileriydi ve bu duygusal verilerin türler ile olan ilişkisinin incelenecekti. Bu konuda yapılmış birçok çalışmanın olduğu ve bu çalışmalarda müziğin farklı açılardan ele alındığı görüldü. Kimi çalışmalar müziğin spektrumları ile ilgilenirken, kimisi nörolojik etkilerini araştırmış ve kimisi de müziğin ses verilerini incelemişti.

Bu çalışmalar detaylı bir şekilde incelendikten sonra, müziğin duygusal verilerinin bulunduğu veri setleri araştırılmaya başlandı. Araştırmalar sonucunda aşağıdaki veri setleri bulunarak incelenmeye alındı:

- MoodsMIREX
- CAL500
- Yang-Dim
- MoodSwings
- NTWICM
- Soundtracks
- DEAP
- AMG1608
- Emotify
- Moodo
- 4Q-emotion
- DEAM/Mediaeval
- PMemo
- Jamendo Moods and Themes
- VGMIDI
- CCMED-WCMED
- Moodo
- MuSe

Bu veri setleri, proje için uygun olan müzik duygusal verilerini içeren veri setlerinin belirlenmesi amacıyla detaylı bir şekilde incelenmeye devam edildi. Bütün bu veri setleri incelendiğinde, projeye olduğu gibi entegre edilecek uygun bir veri seti bulunamadı.

Yine de projenin amacına uygun kullanılabilecek veri setinin MuSe veri seti olduğu belirlendi. Bunun üzerine MuSe (Music Sentiment) veri setinin kullanılmasına karar verildi. MuSe veri seti, 90.001 şarkı için duyarlılık bilgisi içermekte olup, Last.fm aracılığıyla her şarkı için kullanıcının oluşturduğu etiketlere dayalı olarak valence, dominance ve arousal gibi duygusal boyutlar için puanlar hesaplamıştır. Ayrıca, sanatçı, başlık ve tür meta verileri, MusicBrainz Kimliği ve Spotify Kimliği gibi bilgileri içeren ek meta verilerle genişletilebilme özelliğine sahiptir.

2. Veri Setinin Düzenlenmesi ve Temizlenmesi

Ancak MuSe veri seti tek başına yeterli değildi, çünkü önerisi istenecek şarkının duygusal değerleri bilinmiyordu, bu da projenin amacına uymayan bir durumdu. Bu nedenle, veri setindeki Spotify kimliklerinden faydalanma kararı alındı. Spotify Web API'si kullanılarak, veri setindeki her bir şarkının valence, dominance, arousal şeklindeki duygusal özellikleri yerine bu şarkıların Acousticness, Danceability, Energy, Liveness, Loudness, Tempo, Valence gibi neredeyse bütün duygusal özelliklerine erişildi. Böylece veriyi analiz ederken ve modeli eğitirken daha fazla özellik kullanma imkanı sağlanacaktı.

İlk olarak, veri setinde Spotify kimliği bulunmayan şarkılar temizlendi.

A screenshot of a code editor with a dark background and light-colored text. The code is written in Python and uses the pandas library. It consists of eight lines of code, numbered 1 through 8. The code reads a CSV file, drops rows where 'spotify_id' is null, saves the result to a new CSV file, reads the new file, and prints the sum of null values across all columns.

```
1 df = pd.read_csv("musev3-spotify_dataset.csv")
2 df = df.dropna(subset=["spotify_id"])
3 df.to_csv("new_data.csv", index=False)
4
5 df1 = pd.read_csv("musev3-spotify_dataset_new.csv")
6 null_cols = df1.isnull().sum()
7 print(null_cols)
8
```

Şekil 9 Veri setindeki spotify_id'si bulunmayan verileri silme

Ardından Spotify Web API'sine istek gönderilerek, veri setindeki her bir şarkının bahsedilen özellik değerleri çekildi ve bu değerlere göre yeni bir veri seti oluşturuldu.


```

1 # Spotify API connection
2 client_id = "*****"
3 client_secret = "*****"
4
5 client_credentials_manager = SpotifyClientCredentials(client_id, client_secret)
6 sp = spotipy.Spotify(client_credentials_manager=client_credentials_manager)
7
8
9 # Define a function to get the audio features for a track
10 def get_track_genre(track_id):
11     # Get track information from Spotify API
12     track_info = sp.track(track_id)
13     # Get the album information from the track information
14     album_info = track_info["album"]
15     # Try to get the album genres
16     genres = album_info.get("genres")
17     # If the album genres don't exist, get the artist genres
18     if not genres:
19         # Get the list of artist ids for the track
20         artist_ids = [artist["id"] for artist in track_info["artists"]]
21         # Get the list of genres for the artists
22         genres = []
23         for artist_id in artist_ids:
24             artist_info = sp.artist(artist_id)
25             genres += artist_info["genres"]
26         # Remove duplicates from the genres list
27         genres = list(set(genres))
28     # Return the genres
29     return genres
30
31
32 def get_audio_features(track_id):
33     # genres = get_track_genre(track_id)
34     features = sp.audio_features(track_id)[0]
35     return {
36         "acousticness": features["acousticness"],
37         "danceability": features["danceability"],
38         "energy": features["energy"],
39         "liveness": features["liveness"],
40         "loudness": features["loudness"],
41         "tempo": features["tempo"],
42         "valence": features["valence"]
43     }
44     # 'spotify_genre': genres
45
46
47 # Read the CSV file
48 with open("muse_v3_temiz_20k.csv", "r", encoding="utf-8") as f:
49     reader = csv.DictReader(f)
50
51     rows = []
52     i = 0
53     for row in reader:
54         # if i % 100 == 0:
55         print(i)
56         i += 1
57         if i % 250 == 0:
58             time.sleep(10)
59
60         # Get the audio features for the track
61         track_id = row["spotify_id"]
62         features = get_audio_features(track_id)
63
64         # Add the features to the row
65         row.update(features)
66         rows.append(row)
67
68 # Write the updated CSV file
69 fieldnames = reader.fieldnames + list(features.keys())
70 with open("musev3-spotify_dataset_20kSon.csv", "w", encoding="utf-8", newline="") as f:
71     writer = csv.DictWriter(f, fieldnames=fieldnames)
72     writer.writeheader()
73     for row in rows:
74         writer.writerow(row)

```

Şekil 10 Spotify API'ye bağlanma ve gerekli verileri çekip yeni veri seti oluşturma

Oluşturulan veri setinde şu sütun değerleri yer aldı:

- track,
- artist,
- seeds,
- spotify_id,
- genre,
- acousticness,
- danceability,
- energy,
- liveness,
- loudness,
- tempo,
- valence

Bu şekilde, projenin amacına uygun bir veri seti elde edilmiş oldu. Bu veri seti, şarkıların hem duygusal boyutlarını hem de diğer müzik özelliklerini içermesi sayesinde müzik türleri ve duygu durumu arasındaki ilişkiyi yüksek doğruluk oranlarıyla açıklamaya imkan sağladı.

Bu aşamada, bir dizi veri düzenleme işlemi daha gerçekleştirildi. Veri seti üzerinde yapılan işlemler şöyle özetlenebilir:

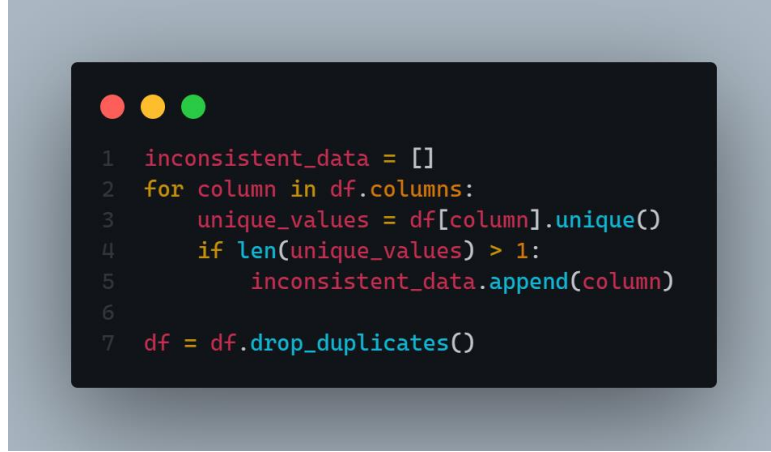
İlk olarak, veri temizleme adımıyla başlandı. Veri temizleme sürecinde, eksik değerlere sahip alanlar tespit edilerek kayıp veri kontrolü yapıldı. Eğer bir veri kayıp ise, ilgili veri setinden çıkarıldı veya eksik değere diğer verilerin ortalaması veya medyan değeri atanarak doldurma işlemi gerçekleştirildi.

A screenshot of a code editor with a dark background and light-colored text. The code is written in Python and is used for data cleaning. It starts with a line that calculates the sum of null values for each column in a DataFrame. Then, it loops through each column. If a column has more than zero null values, it checks the data type. If it's an object type, it drops the column. Otherwise, it calculates the mean of the column and fills the null values with this mean. Finally, it saves the cleaned DataFrame to a CSV file.

```
1 null_cols = df.isnull().sum()
2
3 for column in df.columns:
4     if null_cols[column] > 0:
5         if df[column].dtype == "object":
6             df = df.dropna(subset=[column])
7         else:
8             mean_value = df[column].mean()
9             df[column].fillna(mean_value, inplace=True)
10
11 df.to_csv("temizlenmis_musev3_spotify_dataset.csv", index=False)
```

Şekil 11 Veri temizleme (Eksik değer kontrolü)

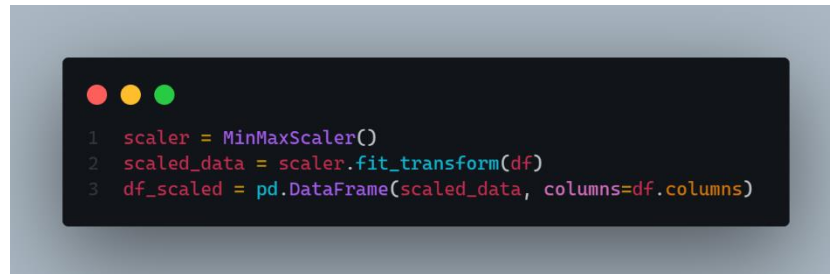
Daha sonra, tutarsız veri kontrolü yapılarak veri setindeki herhangi bir alanın beklenen değerlere uygun olup olmadığı kontrol edildi. Ayrıca, veri setinde tekrarlayan verilerin olup olmadığı da kontrol edildi. Tekrarlayan verilerin bulunması istenmeyen bir durumdur, çünkü modelin bu verilere gereksiz ağırlık vermesine neden olabilir. Bu nedenle, ilgili veriler veri setinden çıkarılarak veri seti temizlendi.

A screenshot of a code editor with a dark background and three colored window control buttons (red, yellow, green) in the top left corner. The code is written in Python and is as follows:

```
1 inconsistent_data = []
2 for column in df.columns:
3     unique_values = df[column].unique()
4     if len(unique_values) > 1:
5         inconsistent_data.append(column)
6
7 df = df.drop_duplicates()
```

Şekil 12 Veri temizleme (Tutarsız veri kontrolü)

Bu aşamalar tamamlandıktan sonra, veri ön işleme adımına geçildi. Veri ön işleme, verinin model için kullanıma hazır hale getirildiği bir aşamadır. Bu aşamada, verinin ölçeklendirilmesi işlemi gerçekleştirildi. Ölçeklendirme, veri setindeki değerleri ortak bir ölçekte normalize etmek anlamına gelir. Bu, modelin daha iyi çalışabilmesi için gereklidir, çünkü genellikle 0 ile 1 arasında belirli bir aralıkta çalışan girdi verilerini daha iyi işleyebilir. Ölçeklendirme işlemi, min-max ölçekleme veya standardizasyon gibi teknikler kullanılarak gerçekleştirilebilir.

A screenshot of a code editor with a dark background and three colored window control buttons (red, yellow, green) in the top left corner. The code is written in Python and is as follows:

```
1 scaler = MinMaxScaler()
2 scaled_data = scaler.fit_transform(df)
3 df_scaled = pd.DataFrame(scaled_data, columns=df.columns)
```

Şekil 13 Veri ön işleme (Min-Max ölçeklendirme)

```
1 scaler = StandardScaler()
2 scaled_data = scaler.fit_transform(df)
3 df_scaled = pd.DataFrame(scaled_data, columns=df.columns)
```

Şekil 14 Veri ön işleme (Standardizasyon ölçeklendirme)

3. Model Eğitimi Denemesi ve Sonuçları

Veri düzenleme ve veri ön işleme adımlarının tamamlanmasının ardından model eğitimi aşamasına geçildi. Bu aşamada öncelikle Lojistik Regresyon, Destek Vektör Makineleri ve Rastgele Orman algoritmaları sırasıyla denendi. Oluşturulan veri seti ile bu algoritmalar kullanılarak gerçekleştirilen model eğitimlerinde accuracy değerinin 0.30 üzerine çıkılmadığı görüldü.

```
1 X_train, X_test, y_train, y_test = train_test_split(
2     X, y, test_size=0.2, random_state=42
3 )
4
5 models = {
6     "Logistic Regression": LogisticRegression(),
7     "Support Vector Machines": SVC(),
8     "Random Forest": RandomForestClassifier(),
9 }
10
11 for name, model in models.items():
12     model.fit(X_train, y_train)
13     y_pred = model.predict(X_test)
14     print(f"{name}:")
15     print("Accuracy:", accuracy_score(y_test, y_pred))
16     print(classification_report(y_test, y_pred, zero_division=1))
17     print("-" * 50)
```

Şekil 15 Lojistik Regresyon, Destek Vektör Makineleri ve Rastgele Orman algoritma modelleri

Accuracy değerini artırmak amacıyla veri setindeki verilerin azaltılmasına karar verildi. Ancak bu işlem sonucunda da sonucun değişmediği görüldü.

```
[Running] python -u "c:\Users\mert8\Desktop\bitirmeTest\raporİçin\modelEğitimAşamaları\classificationTraining1.py"
Logistic Regression:
Accuracy: 0.287524168146364654
-----
Support Vector Machines:
Accuracy: 0.276437984163496468
-----
Random Forest:
Accuracy: 0.380118766468947654
-----
```

Şekil 16 Sınıflandırma modelleri ve sonuçları

Azaltılan veri seti üzerinde PCA kullanılarak yeniden deneme yapıldı, ancak sonucun yine değişmediği gözlemlendi.

```
1  pca = PCA(n_components=2)
2  X_pca = pca.fit_transform(X_scaled)
3
4  data['pca1'] = X_pca[:, 0]
5  data['pca2'] = X_pca[:, 1]
```

Şekil 17 PCA kullanarak boyut indirgeme

Son olarak, veri seti tekrar düzenlendi. Veri setinde yaklaşık olarak 20 farklı genre türü bulunuyordu ancak her bir genre türü eşit sayıda şarkı içermiyordu.

```
[Running] python -u "c:\Users\mert8\Desktop\bitirmeTest\raporİçin\veriDuzenleme\vsGenreyeAitŞarkıSayısıVeAzaltma.py"
genre counts
0  alternative 1490
1  blues 256
2  chill 253
3  classical 269
4  country 392
5  electro 127
6  french 243
7  german 119
8  guitar 126
9  hip-hop 1129
10 indie 5119
11 jazz 943
12 metal 372
13 piano 582
14 pop 3071
15 rap 203
16 reggae 163
17 rock 4220
18 soundtrack 954
19 swedish 113
20 turkish 13
13
```

Şekil 18 İlk eğitilebilir veri seti

Veri seti her bir genre türünde eşit sayıda şarkı bulunacak şekilde yeniden düzenlendi ve aynı algoritmalarla yeniden eğitim denendi.

```
1 balanced_data = pd.DataFrame()
2 for genre in grouped["genre"]:
3     data = df[df["genre"] == genre].sample(n=min_count, random_state=42)
4     balanced_data = pd.concat([balanced_data, data])
5
6 balanced_data.to_csv("duzenlenmismusev3-spotify-dataset.csv", index=False)
```

Şekil 19 Veri setindeki şarkı sayısını en az şarkıya sahip tür sayısına eşitleme

```
[Running] python -u "c:\Users\mert8\Desktop\bitirmeTest\raporİçin\veriDuzenleme\vsGenreyeAitŞarkıSayısıVeAzaltma.py"
genre counts
0 alternative 943
1 hip-hop 943
2 indie 943
3 jazz 943
4 pop 943
5 rock 943
6 soundtrack 943
943
```

Şekil 20 Düzenlenmiş ikinci veri seti

Ancak bahsedilen sınıflandırma algoritmaları ile eğitilen modellerde accuracy değeri hala 0.30 üzerine çıkamadı.

4. Derin Öğrenme Modeline Geçiş ve Sonuçları

Bu noktada modelin derin öğrenme modeline dönüştürülmesi kararı alındı. Literatür araştırmalarında en yüksek accuracy değerinin elde edildiği yöntem bu şekilde olduğu biliniyordu. İlk derin öğrenme modeli tensorflow.keras kütüphanesi kullanılarak oluşturuldu. Sequential modeli, Dense ve Dropout katmanları gibi sınıflar kullanılarak model eğitimi gerçekleştirildi. Veri seti train set ve test set olarak ayrıldı. Özellikler (müzik özellikleri) ve hedef değişken (müzik türü) veri setinden seçildi. Özellik sütunları StandardScaler kullanılarak ölçeklendirildi. Hedef değişken LabelEncoder kullanılarak sayısal bir formata dönüştürüldü.

```

1 feature_columns = ['acousticness', 'danceability',
2                   'energy', 'liveness', 'loudness', 'tempo', 'valence']
3 X = data[feature_columns]
4 y = data['genre']
5
6 X_train, X_test, y_train, y_test = train_test_split(
7     X, y, test_size=0.2, random_state=42)
8
9 scaler = StandardScaler()
10 X_train = scaler.fit_transform(X_train)
11 X_test = scaler.transform(X_test)
12
13 encoder = LabelEncoder()
14 y_train = encoder.fit_transform(y_train)
15 y_test = encoder.transform(y_test)

```

Şekil 21 Veri seti özelliklerini belirleme, hedef değişken belirleme, veri setini train-test olarak ayırma, ölçeklendirme ve sayısal formata dönüştürme

Böylece derin öğrenme modeli (Sequential) oluşturuldu. Modelde, tam bağlantılı (Dense) katmanlar ve aşırı öğrenme (overfitting) karşı koruma sağlamak için Dropout katmanları kullanıldı. Kayıp fonksiyonu olarak ortalama karesel hata (mean_squared_error) ve optimize edici olarak adam kullanıldı. Özetle, çok katmanlı yapay sinir ağları kullanılarak model eğitimi gerçekleştirildi. Ancak accuracy değeri hala beklenen seviyede değildi.

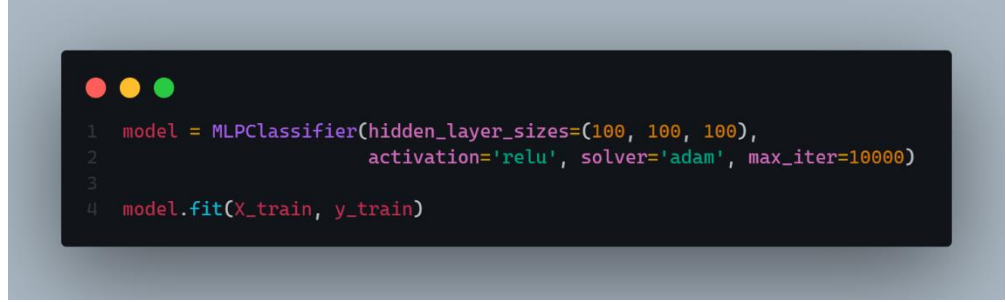
```

1 model = Sequential()
2 model.add(Dense(64, activation="relu", input_dim=X_train.shape[1]))
3 model.add(Dropout(0.5))
4 model.add(Dense(32, activation="relu"))
5 model.add(Dropout(0.5))
6 model.add(Dense(len(set(y_train)), activation="softmax"))
7
8 model.compile(loss="mean_squared_error", optimizer="adam", metrics=["accuracy"])
9
10 model.fit(
11     X_train,
12     y_train_one_hot,
13     epochs=50,
14     batch_size=16,
15     validation_data=(X_test, y_test_one_hot),
16 )
17

```

Şekil 22 Çok katmanlı model eğitimi denemesi

Bunun üzerine farklı bir çok katmanlı yapay sinir ağı modeli kullanma kararı alındı. Bu sefer sklearn.neural_network kütüphanesinden MLP sınıflandırıcısı kullanılmak istendi. Veriler okundu, özellik sütunları ve hedef değişken belirlendi. Veriler StandardScaler kullanılarak ölçeklendirildi. MLPClassifier sınıfı kullanılarak model oluşturuldu. Gizli katmanlarda 100 nöron ve 'relu' aktivasyon fonksiyonu kullanıldı ve optimizasyon algoritması olarak 'adam' seçildi. Ancak accuracy değerlerinin hala istenen seviyede olmadığı ve genellikle 0.25 ile 0.35 arasında değiştiği görüldü.



```
1 model = MLPClassifier(hidden_layer_sizes=(100, 100, 100),
2                       activation='relu', solver='adam', max_iter=10000)
3
4 model.fit(X_train, y_train)
```

Şekil 23 Çok katmanlı model eğitimi denemesi (MLP Classifier)

5. Doğru Modeli Bulma ve Sonuç

Son olarak, farklı bir derin öğrenme yöntemini denemeye karar verildi. Bir MLP modeli oluşturuldu yani katman sayıları artırıldı. Bu modeli oluşturmak için Keras'ın fonksiyonel API'si kullanıldı. Modelin katmanları şu şekildedir:

- Giriş katmanında tf.keras.layers.Dense ile bir tam bağlantılı (fully connected) katmana geçilir. Bu katman 256 nöron içerir ve ReLU aktivasyon fonksiyonunu kullanılır. Bu sayede, giriş verileri üzerinde doğrusal olmayan dönüşümler gerçekleştirilir.
- İkinci katman, aşırı öğrenme (overfitting) karşı koruma sağlamak için tf.keras.layers.Dropout kullanılarak oluşturulur. Dropout, ağırlıkları rastgele olarak sıfıra atarak aşırı öğrenmeyi azaltmaya yardımcı olan bir düzenleme tekniğidir.
- Üçüncü, dördüncü ve beşinci katmanlar da tam bağlantılı katmanlardır (tf.keras.layers.Dense). Sırasıyla 128, 64 ve 32 nöron içerirler ve ReLU aktivasyon fonksiyonunu kullanırlar. Bu katmanlar, ağına daha karmaşık özellikler öğrenmesine yardımcı olur.
- Çıkış katmanı, sınıflandırma problemini çözmek için kullanılır. Bu örnekte 17 sınıf olduğu için tf.keras.layers.Dense kullanılarak oluşturulan çıkış katmanı 17 nörona sahiptir ve softmax aktivasyon fonksiyonunu kullanır. Softmax, çıkış değerlerini olasılık dağılımına dönüştürerek sınıflandırma yapılmasını sağlar.

Model, `tf.keras.Model` sınıfından türetilir ve giriş ve çıkış katmanları belirtilerek oluşturulur.

Modelin derleme adımında 'adam' optimizasyon algoritması kullanılmıştır ve kayıp fonksiyonu olarak 'sparse_categorical_crossentropy' tercih edilmiştir.

Ayrıca, eğitim sırasında doğruluk metriği olarak "accuracy" kullanılır.

Son olarak model eğitiminde veriler (`X_train`) ve etiketler (`y_train`) verilir ve 1500 epoch süresince eğitim yapılır. Aynı zamanda, modelin performansını değerlendirmek için doğrulama verisi (`X_test` ve `y_test`) de kullanılmıştır.

Bu şekilde oluşturulan modelin çalıştırılmasıyla 0.60'a kadar accuracy değerleri elde edilmiştir. Epoch sayısının artırılmasıyla ise accuracy değeri 0.94'e kadar çıkmıştır.

```
1 inputs = tf.keras.Input(shape=(7,))
2 x = tf.keras.layers.Dense(256, activation="relu")(inputs)
3 x = tf.keras.layers.Dropout(0.2)(x)
4 x = tf.keras.layers.Dense(128, activation="relu")(x)
5 x = tf.keras.layers.Dense(64, activation="relu")(x)
6 x = tf.keras.layers.Dense(32, activation="relu")(x)
7 outputs = tf.keras.layers.Dense(17, activation="softmax")(x)
8
9 model.compile(
10     optimizer="adam", loss="sparse_categorical_crossentropy", metrics=["accuracy"]
11 )
12
13 model.fit(X_train, y_train, epochs=1500, validation_data=(X_test, y_test))
```

```
Epoch 2998/3000
165/165 [=====] - 0s 1ms/step - loss: 0.1575 - accuracy: 0.9504 - val_loss: 7.9934 - val_accuracy: 0.3323
Epoch 2999/3000
165/165 [=====] - 0s 1ms/step - loss: 0.1583 - accuracy: 0.9534 - val_loss: 8.0225 - val_accuracy: 0.3278
Epoch 3000/3000
165/165 [=====] - 0s 1ms/step - loss: 0.1581 - accuracy: 0.9479 - val_loss: 7.9446 - val_accuracy: 0.3210
```

Şekil 24 Katman sayıları artırılmış, epoch değeri yükseltilmiş MLP modeli ve bu modelin sonuçları

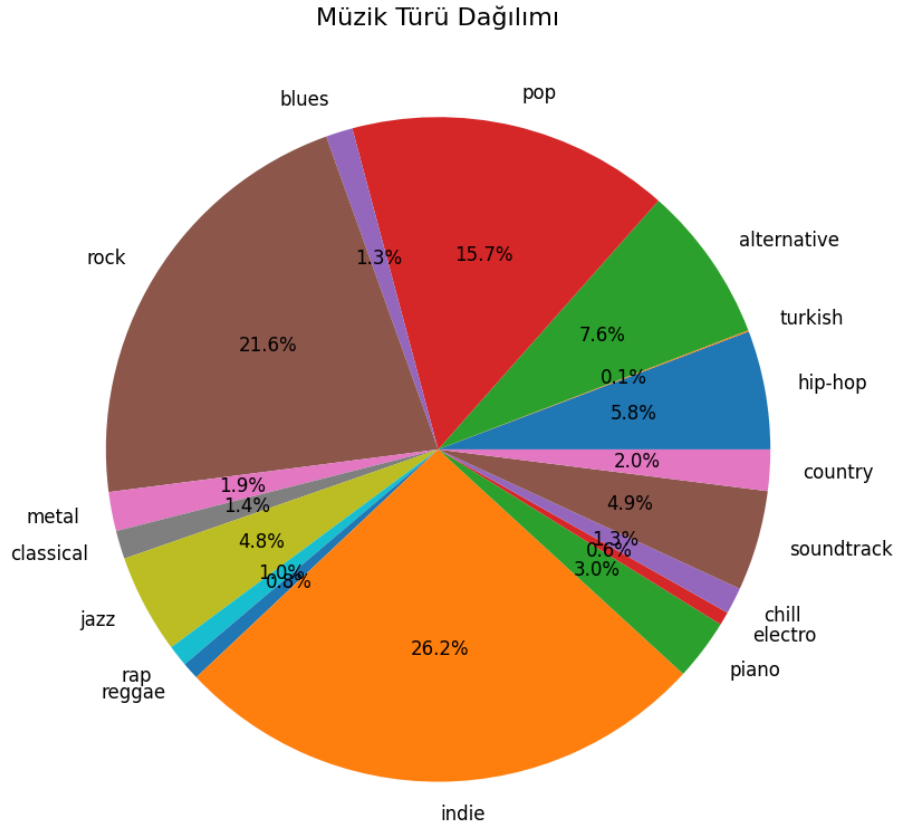
Alınan sonuçların başarılı olduğu görüldü fakat projedeki veri seti çeşitliliğini artırma amacıyla veri setinde değişiklik yapıldı. Önceki model eğitimlerinde accuracy değerini artırmak amacıyla müzik türü sayısı düşürülmüş, buna paralel olarak müzik verisi sayısı da düşürülmüştür. Hatta her bir müzik türüne ait eşit sayıda müzik verisinin olması amaçlanmış ve bu yüzden veri setindeki toplam müzik verisi sayısı 6600'e kadar gerilemiştir. Bu kadar az verinin olmasının proje için verimli olmayacağına karar verilmiş ve önceki veri setine dönüş yapılmıştır. Bu veri setinde şarkı sayısı ve şarkı türü çeşitliliği çok fazladır.

```
[Running] python -u "c:\Users\mert8\Desktop\bitirmeTest\raporİçin\veriDüzenleme\vsGenreyeAitŞarkıSayısıVeAzaltma.py"
genre counts
0 alternative 1490
1 blues 256
2 chill 253
3 classical 269
4 country 392
5 electro 127
6 hip-hop 1129
7 indie 5119
8 jazz 943
9 metal 372
10 piano 582
11 pop 3071
12 rap 203
13 reggae 163
14 rock 4220
15 soundtrack 954
16 turkish 13
13
```

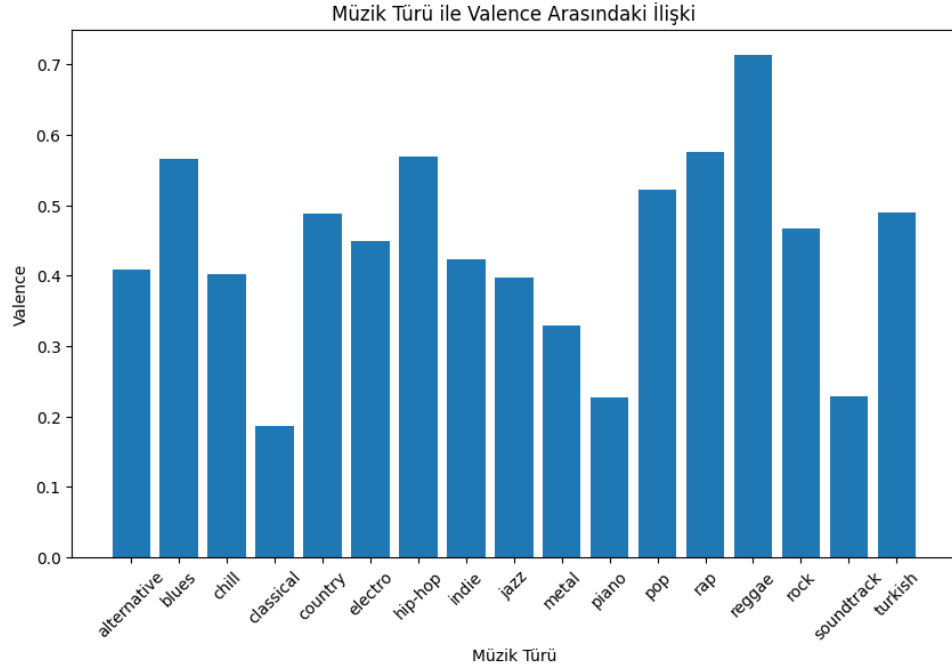
Şekil 25 Düzenlenmiş ve müzik türü artırılmış en son veri seti (19558 veri)

Model, bu veri setiyle yeniden eğitildi ve 0.65 accuracy değeri elde edildi, bu değer kabul edilebilir bir sonuç olarak değerlendirildi. Benzer çalışmalarda da bu doğruluk oranlarına ulaşıldığı görüldü.

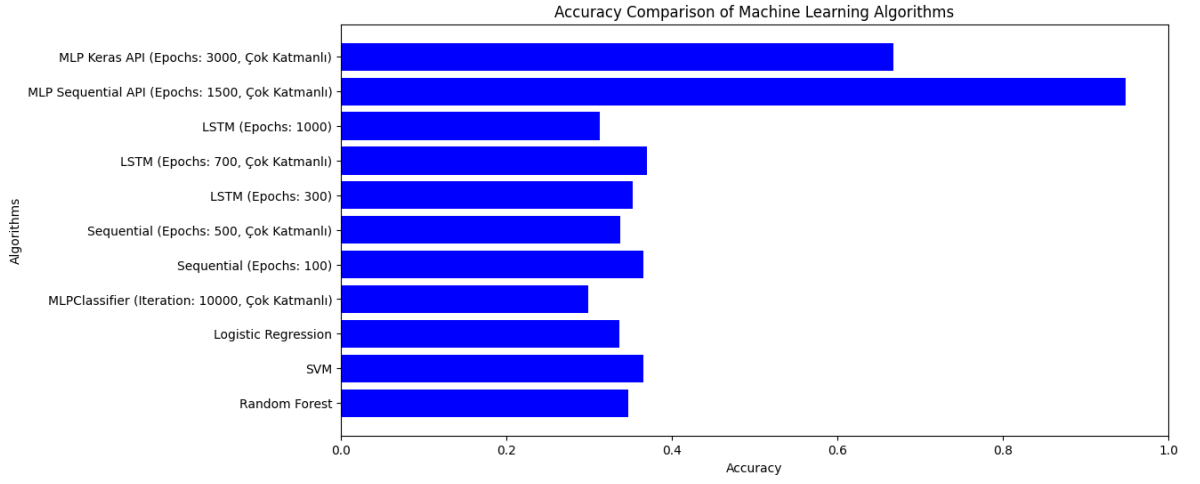
En son kullanılan veri setinin müzik verilerinin tür dağılımı, bu müzik türleri ile “valence” özelliği arasındaki ilişki ve model eğitiminde kullanılan algoritmaların accuracy değerleri aşağıda belirtilmiştir.



Şekil 26 En son veri setinin tür dağılımı



Şekil 27 Veri setindeki müzik verilerinin valence değerleri ile müzik türleri arasındaki ilişki



Şekil 28 Model eğitiminde kullanılan algoritma ve yöntemlerinin accuracy değerleri

6. Eğitilen Modeli Kaydetme ve Öneri Sistemine Geçiş

Model eğitildikten sonra, ilgili model bir klasöre kaydedildi ve statik olarak öneri sistemi denemelerine başlandı. Öneri sistemine geçmeden tahmin etme sistemi denendi. Bu sistemde önceden eğitilen model kullanılarak static olarak verilen belirli bir müziğin özelliklerine göre o müziğin hangi tür olduğuna dair tahminler yapılmaya başlandı. Bu tahminler yapılırken klasör içerisinde kaydedilmiş eğitilmiş model kullanılıyordu.



Şekil 29 Modeli kaydetme ve tahmin sistemi oluşturma

7. Öneri Sistemini Oluşturma

Projeyi öneri sistemi olarak kullanılabilir hale getirilmek amacıyla gerekli adımlar takip edildi. İlk olarak, benzerlik hesaplamaları için bir model oluşturuldu ve eğitildi. Ardından, cosine_similarity yöntemi kullanılarak müzik verileri ile veri setindeki diğer şarkılar arasındaki benzerlikler hesaplandı.

Bu adımlar tamamlandıktan sonra, verilen bir şarkı için benzerlik skorları ve indeksler sıralandı. En yakın 5 şarkının track, artist, spotify_id, seeds ve genre gibi istenen sütunları ile benzerlik yüzdesini (similarity_percentage) içeren bir liste oluşturuldu. Bu sayede, statik olarak verilen bir şarkıya en yakın 5 şarkının önerileri elde edildi.

```

1  muzik = [0.209,0.637,0.678,0.156,-3.798,84.039,0.254]
2
3  muzik_df = pd.DataFrame([muzik], columns=X.columns)
4  muzik_scaled = scaler.transform(muzik_df)
5
6  similarities = cosine_similarity(muzik_scaled, X_scaled)
7
8  sorted_similarity_indices_and_values = sorted(
9      enumerate(similarities[0]), key=lambda x: x[1], reverse=True
10 )
11
12  recommended_songs = []
13
14  for index, similarity in sorted_similarity_indices_and_values[1:6]:
15      song_info = df[["track", "artist", "spotify_id", "seeds", "genre"]].iloc[index]
16      similarity_percentage = similarity * 100
17      recommended_songs.append(
18          {
19              "Benzerlik Oranı": f"{similarity_percentage:.2f}%",
20              "Track": song_info["track"],
21              "Artist": song_info["artist"],
22              "Spotify ID": song_info["spotify_id"],
23              "Seeds": song_info["seeds"],
24              "Genre": song_info["genre"],
25              "cover": get_track_photo(song_info["spotify_id"])
26          }
27      )
28  return recommended_songs

```

Şekil 30 Kaydedilen model yardımıyla öneri sistemi oluşturma

Daha sonra, sistemin dinamik hale getirilebilmesi için, kullanıcının girdiği şarkının kimliğiyle birlikte Spotify Web API'den ilgili şarkının özellik değerleri çekildi. Bu işlemde spotipy kütüphanesi ve Spotify API'si kullanıldı. Şarkının özellik değerleri, Spotify API'den alınarak, eğitilmiş modele girdi olarak verildi ve veri setinde bu şarkıya benzeyen 5 şarkı liste haline getirildi.



Şekil 31 Spotify API yardımı ile öneri sistemini dinamik hale getirme

Bu şekilde, dinamik bir öneri sistemi oluşturulmuş oldu. API'ye erişim sağlanarak kullanıcının girdiği şarkıya en yakın 5 şarkının önerileri elde edildi ve eğitilmiş model kullanılarak benzerlik hesaplamaları yapıldı.

8. Sistemin Web Platformuna Entegrasyonu

Son aşamada, bu sistem bir web platformuna dönüştürüldü. Model eğitimi sırasında Python kullanıldığından, uygulamanın backend kısmında Django kullanmaya karar verildi. Backend kısmında yalnızca bir POST işlemi gerçekleştirilmekte ve bu işlem, frontend tarafından gelen Spotify şarkı linkindeki id kısmını alarak modele iletilip bir sonuç döndürmektedir.

```
1 @api_view(["POST"])
2 def music_recommendation(request):
3     try:
4         spotify_link = request.data["spotify_link"]
5         print("Received data:", request.data)
6         pattern = r"track/(\w+)"
7         track_id = re.search(pattern, spotify_link)
8         link = ""
9         if track_id:
10             link = track_id.group(1)
11         else:
12             return Response({
13                 "status": False,
14                 "message": "Spotify ID bulunamadı."
15             })
16         tahmin, recommended_songs = music_similarity(get_audio_features(link))
17         return Response({
18             "status": True,
19             "genre": tahmin,
20             "recommended_songs": recommended_songs
21         })
22     except Exception as ex:
23         return Response({
24             "status": False,
25             "message": ex
26         })
```

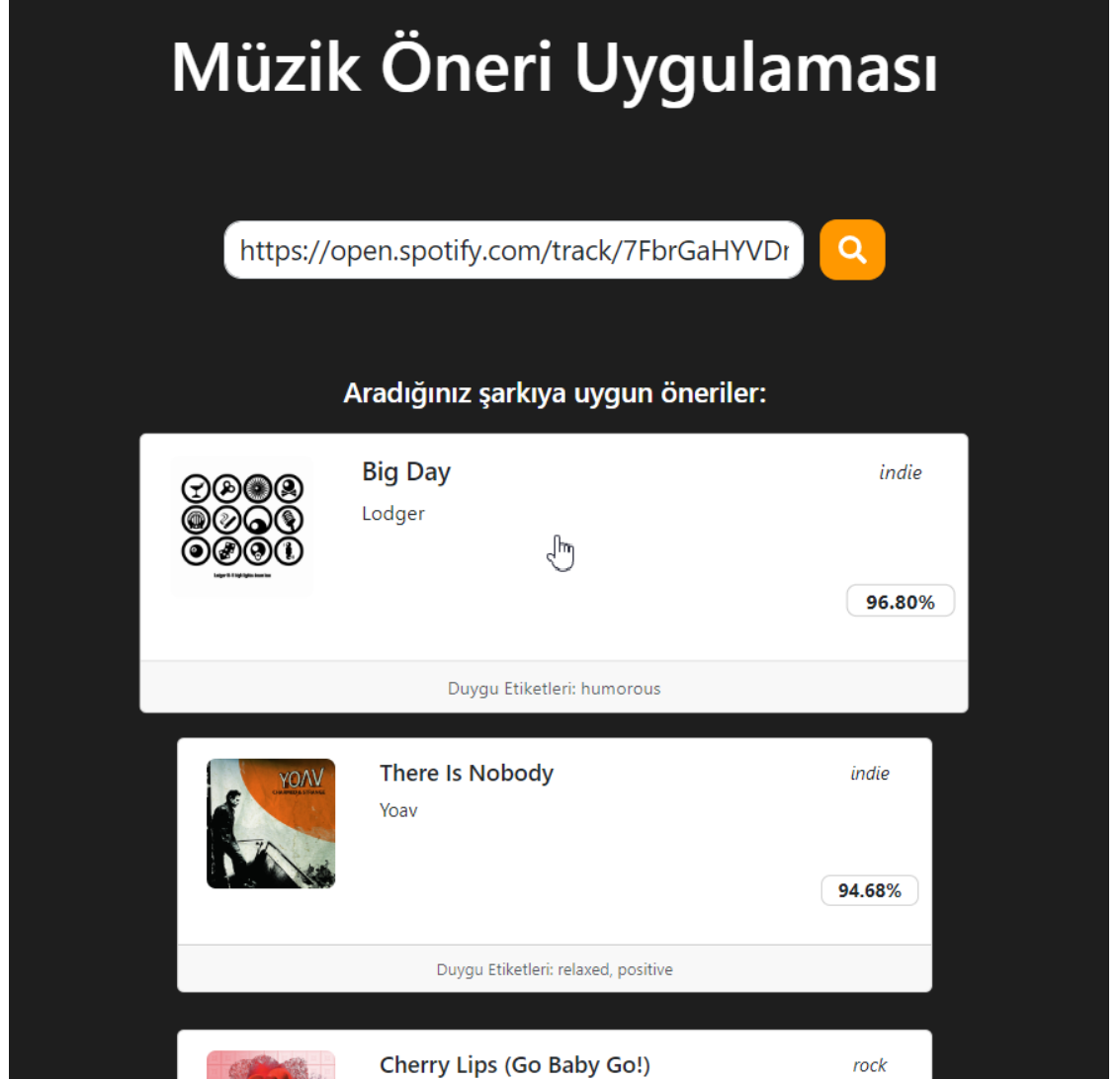
Şekil 32 Öneri sistemi backend yapısı

Frontend kısmında ise React ve Bootstrap tercih edildi. Bu kısımda başlık, bir input alanı ve bu inputu göndermek için bir buton eklenerek kullanıcı arayüzü oluşturuldu.



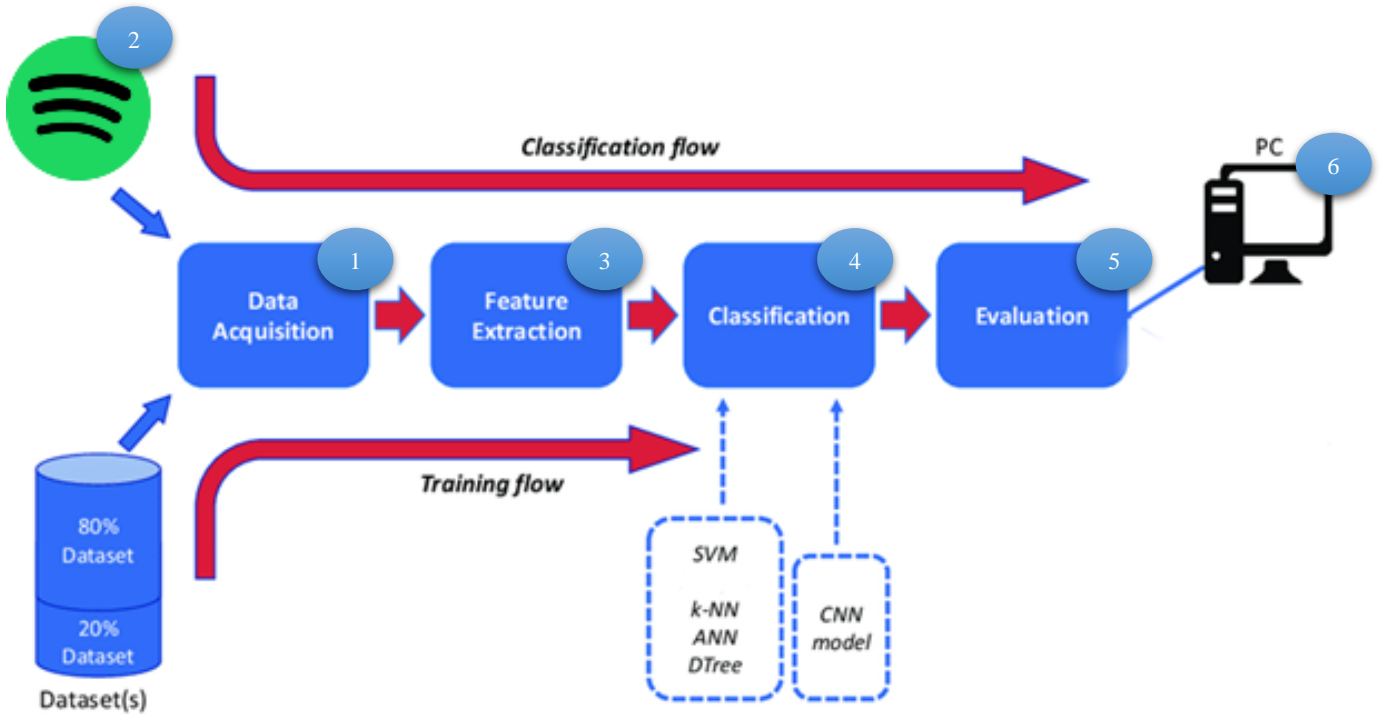
Şekil 33 Web Platformu şarkı girişi

Web platformunda kullanıcı, Spotify şarkı linkini girerek ilgili şarkının türüne benzer 5 şarkıyı öneri olarak alabilmektedir. Backend tarafında, kullanıcının girdiği şarkının Spotify API'ye yapılan isteğe bağlı olarak elde edilen özellik değerlerini model ile karşılaştırarak benzerlik analizini gerçekleştirebilmektedir. Elde edilen sonuçlar frontend tarafına iletilerek kullanıcıya gösterilmektedir. Bu şekilde, kullanıcılar web platformu üzerinden müzik türlerine göre öneriler alabilmektedir.



Şekil 34 Web platformu şarkı önerisi

Böylece proje tamamlanmış oldu. Artık kullanıcı spotify linkini bildiği bir şarkıyı bu siteye girip arattığında, müzikal veriler ve duygusal veriler açısından bu şarkıya en çok benzeyen şarkıları listeleyebilecek. Listelediği şarkının ne kadar benzediğini ve bu şarkıların duygusal etiketlerini görebilecek ve isterse anında bu şarkıları dinleyebilecek.



Şekil 35 Proje Özeti Diyagramı

1. Veri seti okunur. Okunan veri; veri temizleme, tutarsız veri kontrolü gibi işlemlerden geçerek ön işleme için hazırlanır.
2. Ön işleme adımından geçmiş veri seti ile Spotify API'den alınan özellikler ile veri seti tekrardan oluşturuldu.
3. Yeniden oluşturulmuş veri seti; normalizasyon, standardizasyon gibi teknikler ile ölçeklendirilir ve veri model için kullanıma hazır hale getirilir
4. Eğitilmek için kullanılacak model belirlenir. Modeller belirlenirken veri setinin yapısı ve performans değeri göz önünde bulundurulur.
5. Belirlenmiş modeller ile veri seti eğitilir. Performans sonuçlarına göre model geliştirilir, farklı model tipleri denenir.
6. Eğitilmiş ve çalışan model seti, Web Platformuna entegre edilir ve kullanıcının verdiği müziği analiz edip kullanıcıya veri setinden benzer veriler öneren bir sistem oluşturulur.

5. TARTIŞMA VE SONUÇ

Bu proje, yapay sinir ağları kullanılarak müzik türlerinin sınıflandırılması ve kullanıcı duygu durumu analizi için bir öneri sistemi geliştirme amacıyla gerçekleştirilmiştir. Projede kullanılan yöntemler ve elde edilen sonuçlar göz önüne alındığında, projenin ana hedeflerine ulaşıldığı söylenebilir. Kullanıcının dinlediği şarkının türünün sınıflandırılması ve duygu durumunun analizi, yüksek doğruluk oranlarıyla gerçekleştirilmiştir. Ayrıca, kullanıcının duygu durumu ve türüne benzer şarkılar öneren bir öneri sistemi başarıyla oluşturulmuştur.

Proje sürecinde, uygun bir veri seti seçimi büyük önem taşımaktadır. İlk olarak, GTZAN veri seti düşünülmüş olsa da müziklerin tam şarkı verileri olmaması nedeniyle projeye uygun değildir. Bu durumda, MuSe (Music Sentiment) veri seti kullanılarak çalışmalar devam etmiştir. MuSe veri seti, 90.001 şarkı için duyarlılık bilgisi içermekte ve kullanıcı tarafından oluşturulan etiketlere dayalı olarak duygusal boyutları için puanlar hesaplamaktadır. Spotify Web API kullanılarak bu verilere ek özellikler de eklenmiştir. Bu şekilde daha kapsamlı bir veri seti elde edilmiştir.

Model eğitimi aşamasında, farklı algoritmalar ve derin öğrenme modelleri kullanılarak çalışmalar gerçekleştirilmiştir. İlk denemelerde elde edilen sonuçlar beklenen doğruluk seviyesine ulaşmamıştır. Ancak, modelin katmanlarının ve özelliklerin düzenlenmesiyle birlikte doğruluk oranlarında artış görülmüştür. Özellikle, geliştirilen Keras API MLP modeliyle yüksek doğruluk değerlerine ulaşılmıştır. Bu model, giriş katmanı, gizli katmanlar ve çıkış katmanı olmak üzere farklı katmanlardan oluşmaktadır.

Projenin bir sonraki adımında, geliştirilen modelin bir web platformuna dönüştürülmüştür. Bu amaçla, arka yüz tarafında Django frameworkü kullanılarak bir API oluşturulmuş ve ön yüz tarafında React ve Bootstrap kullanılmıştır. Kullanıcılar, web platformu üzerinden Spotify şarkı linkini girerek duygusal durumu ve türüne benzer şarkıları öneri olarak alabilmektedir.

Projede ayrıca daha çok Türkçe müzik verisinin projeye entegre edilmesi hedeflenmektedir. Bu sayede, Türkçe müzik verilerine ait bir veri seti oluşturularak daha spesifik ve lokal bir analiz yapılması amaçlanmaktadır. Türkçe müzik verilerine ait veri seti oluşturulmasıyla birlikte bu alanda yapılan çalışmalara katkı sağlanabilecektir.

Sonuç olarak bu projede müzik türlerinin sınıflandırılması ve duygu durumu analizine dayalı bir öneri sistemi başarıyla geliştirilmiştir. Proje hedefleri doğrultusunda, kullanıcıların dinledikleri şarkılara göre duygusal durumlarını analiz edebilmeleri ve benzer türde şarkıları keşfedebilmeleri sağlanmıştır. Projede kullanılan yapay sinir ağı modelleri ve algoritmalarıyla yüksek doğruluk değerlerine ulaşılmıştır. Öneri sistemi bir web platformuna dönüştürülerek kullanıcı dostu bir arayüz sunulmuştur.

KAYNAKLAR

1. Özdemir, Ö. F., & Gümüş, İ., 2017, A review on music genre classification methods, Turkish Journal of Electrical Engineering & Computer Sciences, 25(2), 413-432.
2. Li, L., & Lee, C., 2018, A survey of music genre classification using machine learning techniques, Journal of Ambient Intelligence and Humanized Computing, 9(2), 197-212.
3. Kočí, R., & Schuller, B., 2015, Automatic music genre classification: A comprehensive survey, ACM Computing Surveys (CSUR), 47(1), 1-35.
4. Chuan, C., & Chai, Y., 2017, Music genre classification using artificial neural networks, In 2017 International Conference on Computer and Information Science (ICCIS) (pp. 35-40). IEEE.
5. Liu, Y., & Li, J., 2017., Music genre classification based on an improved convolutional neural network., Multimedia Tools and Applications, 76(16), 17261-17276.
6. Doğan Ö., 23/12/2022, Makine Öğrenmesi Nedir? – Makine Öğrenmesi Algoritmaları [online], <https://teknoloji.org/makine-ogrenmesi-nedir-makine-ogrenmesi-algoritmaları/>
7. Akca F., 23/12/2022, Nedir Bu Destek Vektör Makineleri? (Makine Öğrenmesi Serisi-2) [online], <https://medium.com/deep-learning-turkiye/nedir-bu-destek-vekt%C3%B6r-makineleri-makine-%C3%B6%C4%9Frenmesi-serisi-2-94e576e4223e>,
8. Ocak M., 23/12/2022, Yapay Sinir Ağları Nedir? [online], <https://bilimgenc.tubitak.gov.tr/yapay-sinir-aglari-nedir>
9. Ergin T., 23/12/2022, Convolutional Neural Network (ConvNet yada CNN) nedir, nasıl çalışır? [online], <https://medium.com/@tuncerergin/convolutional-neural-network-convnet-yada-cnn-nedir-nasil-calisir-97a0f5d34cad>
10. Alicja W., Piotr S., Zbigniew W. R., 04/02/2023, Multi-Label Classification of Emotions in Music [online], https://link.springer.com/chapter/10.1007/3-540-33521-8_30
11. Arsh C., 12/02/2023, Music Genre Classification Using CNN [online], <https://www.clairvoyant.ai/blog/music-genre-classification-using-cnn>
12. Raghav A., 22/02/2023, Music Genre Classification Project Using Machine Learning Techniques [online], <https://www.analyticsvidhya.com/blog/2022/03/music-genre-classification-project-using-machine-learning-techniques/>
13. Omair A., 02/03/2023, Music Genre Classification Project using Deep Learning [online], <https://www.projectpro.io/article/music-genre-classification-project-python-code/566>
14. Karen De la R., 13/03/2023, Machine Learning analysis [online], <https://rpubs.com/kdrdatascience/862303>
15. Nagesh Singh C., 21/03/2023, Audio Data Analysis Using Deep Learning with Python [online], <https://www.kdnuggets.com/2020/02/audio-data-analysis-deep-learning-python-part-1.html>

16. Andrada A., 30/03/2023, GTZAN Dataset - Music Genre Classification [online], <https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification>
17. Andrada A., 05/04/2023, Work w/ Audio Data: Visualise, Classify, Recommend [online], <https://www.kaggle.com/code/andradaolteanu/work-w-audio-data-visualise-classify-recommend/input>
18. Juansgomez87(Github), 07/04/2023, Repository collects information about different data sets for Music Emotion Recognition. [online], https://github.com/juansgomez87/datasets_emotion#ccmed-wcmed
19. Christopher A., Manuel B., 10/04/2023, MuSe: The Musical Sentiment Dataset [online], <https://openhumanitiesdata.metajnl.com/articles/10.5334/johd.33#3-dataset-description>
20. Deniz K., 15/04/2023, Python ile Veri Ön İşlemeye Dalış [online], <https://medium.com/@denizkilinc/python-ile-veri-%C3%B6n-i%C5%9Flemeye-dal%C4%B1%C5%9F-f89f921658bd>
21. Cristóbal V., 20/04/2023, Predicting the Music Mood of a Song with Deep Learning [online], <https://towardsdatascience.com/predicting-the-music-mood-of-a-song-with-deep-learning-c3ac2b45229e>
22. Spotify, 01/05/2023, Get Track's Audio Features [online], <https://developer.spotify.com/documentation/web-api/reference/get-audio-features>
23. Mohan R., 07/05/2023, Building a React.js Application using Django REST Framework [online], <https://www.section.io/engineering-education/react-and-django-rest-framework/>
24. MLJAR, 15/05/2023, Machine Learning with Django [online], <https://www.deploymachinelearning.com/>

ÖZGEÇMİŞ

Mert Yılmaz

2001 Amasya doğumlu. İlkokulu İbrahim Yapıcı İlköğretim okulunda tamamladı. Ortaokulu Özel Diltaş Ortaokulu'nda, liseyi de Özel Diltaş Anadolu Lisesi'nde tamamladı. 2019 yılından beri İstanbul Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği bölümüne devam etmektedir.

EKLER

EK-C

Talimat Alma-Verme Çizelgesi

Takım Liderinin Adı Soyadı: Mert Yılmaz	
	Puanlama
	Mert Yılmaz
i. Talimatların Açık ve Anlaşılır Olması	
ii. Doğru Talimatların Verilmiş Olması / Talimatların Gerekliliği	
iii. Talimatların Uygulanması / Talimatların Gerçekleşmesi	
iv. Yeterli Sayıda Talimat Verilmesi	
v. Talimatların Doğru Bir Üslupla Verilmiş Olması	