T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ONTOLOJİ VE DURUMSAL TABANLI ÇIKARSAMAYA DAYALI İŞBİRLİKÇİ TAVSİYE YÖNTEMİ

ZAFER DÜZEN

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ PROGRAMI

DANIŞMAN
YRD. DOÇ. DR. MEHMET S. AKTAŞ

İSTANBUL, 2016

T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

ONTOLOJİ VE DURUMSAL TABANLI ÇIKARSAMAYA DAYALI İŞBİRLİKÇİ TAVSİYE YÖNTEMİ

Zafer DÜZEN tarafından hazırlanan tez çalışması 19.07.2016 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Yrd. Doç. Dr. Mehmet S. AKTAŞ

Yıldız Teknik Üniversitesi

Jüri Üyeleri

Yrd. Doç. Dr. Mehmet S. AKTAŞ

Yıldız Teknik Üniversitesi _____

Doç. Dr. Songül ALBAYRAK

Yıldız Teknik Üniversitesi _____

Doç. Dr. Çağatay ÇATAL

İstanbul Kültür Üniversitesi _____

Bu çalışmaya maddi destek Tübitak 114E781 nolu ve “Provenans Kullanılarak Sosyal Medya Yazılımlarında Bilgi Kirliliğinin Ve Telif Hakları İhlallerinin Tespiti İçin Yöntemler Geliştirilmesi” başlıklı projesi tarafından sağlanmıştır.

ÖNSÖZ

Günümüzde artan veri yoğunluğuyla birlikte kullanıcıların bilgiye kolay ulaşımını sağlamak için bu veriler anlamlı hale getirilmeye çalışılmaktadır. Öneri sistemleri de anlamlı veri çıkarımının yoğunlukla kullanıldığı alanlardan biri olmaktadır.

Bu çalışmada LastFM adlı müzik paylaşım sitesinden elde edilen verilerin işlenmesiyle kullanıcılara öneri yapabilen bir uygulama oluşturulmuştur. Oluşturulan öneri motoru için durum tabanlı çıkarsama ve ontolojik veri zenginleştirme ile hibrit bir yöntembilim geliştirilmiş ve işbirlikçi filtreleme yönteminin dezavantajları çözülmeye çalışılmıştır.

Çalışma süresince bilgisi, yönlendirmeleri ve pozitif yaklaşımı ile bana büyük destek olan değerli hocam Yrd. Doç. Dr. Mehmet S. AKTAŞ'a sonsuz teşekkürlerimi sunarım. Ayrıca çalışmamda ve hayatımın her alanında desteklerini esirgemeyen eşime çok teşekkür ederim.

Temmuz, 2016

Zafer DÜZEN

İÇİNDEKİLER

Sayfa

SİMGE LİSTESİ viii

KISALTMA LİSTESİ ix

ŞEKİL LİSTESİ x

ÇİZELGE LİSTESİ xii

BÖLÜM 1

GİRİŞ 1

1.1 Literatür Özeti 1
1.2 Tezin Amacı 1
1.3 Hipotez 2

BÖLÜM 2

ARAŞTIRMA PROBLEMİ 3

2.1 Problemin Tanımı 3

BÖLÜM 3

TEMEL BİLGİLER ve LİTERATÜR ÇALIŞMASI 7

3.1 Temel Bilgiler 7
3.2 İşbirlikçi Filtreleme Metodolojisi 7
3.3 Durum Tabanlı Çıkarsama Metodolojisi 9
3.4 Ontoloji 12
3.5 Literatür Çalışması 13
3.6 İşbirlikçi Filtreleme Teknikleri Üzerindeki Kaynak Çalışması 13
3.7 Bilgi Tabanlı Sistemler Üzerindeki Kaynak Çalışması 14
3.8 İçerik Tabanlı Filtreleme Sistemleri Üzerindeki Kaynak Çalışması 14
3.9 Hibrit Tavsiye Sistemleri Üzerindeki Kaynak Çalışması 15
3.10 Hibrit Yaklaşımlardaki Eksiklikler 16

3.11 Tez Kapsamındaki Öneri 17

BÖLÜM 4

METODOLOJİ 18

4.1 Uygulama Alanı 18
4.2 Veri Seti 19
4.3 Önermekte Olunan Tavsiye Sistemindeki İş Akışı 19
4.4 Önermekte Olunulan Tavsiye Sisteminin Modüler Bileşenleri 23
4.4.1 CBR Durum Ontolojisi 23
4.4.2 Reasoner 23
4.4.3 RDF Grafiği 23
4.4.4 RDF Modülü 23
4.4.5 CBR Motoru 24
4.4.6 CF Motoru 24
4.4.7 Hibrit Yöntem Bileşeni 24
4.4.8 Kullanıcı Geri Bildirim Modülü 24
4.4.9 Geri Bildirim Analiz Bileşeni 24
4.4.10 CF Veri Tabanı 25
4.4.11 CBR Veri Tabanı 25
4.5 CBR Veri Tanımlama Modeli 25
4.6 Durum Tabanlı Çıkarsama (Inference) Yeteneği 32
4.7 Ağırlıklandırma Yaklaşımı 34
4.8 Hibrit Yaklaşım 35
4.9 Hibrit Yaklaşım Algoritması 35

BÖLÜM 5

SİSTEM MİMARİSİ 38

5.1 Veri Katmanı 39
5.2 Veri Erişimi Katmanı 39
5.3 Uygulama Katmanı 40
5.4 Kullanıcı Ara Yüzü Katmanı 40

BÖLÜM 6

UYGULAMA DETAYLARI 41

BÖLÜM 7

DENEY SONUÇLARI ve DEĞERLENDİRME 50

7.1 Deneysel Sonuçlar 51
7.1.1 Kullanıcı - Etiket Öneri Sonuçları 51
7.1.2 Kullanıcı - Şarkı Öneri Sonuçları (Tek Ülke - İngiltere) 62
7.1.3 Tüm Şarkılar Üzerinden Yapılan Öneri Sonuçları 74

7.2 Değerlendirme 86
7.2.1 Araştırma Sorularına Verilen Yanıtlar 86

BÖLÜM 8

SONUÇ ve ÖNERİLER 89

EK-A

MÜZİK ONTOLOJİSİ 93

ÖZGEÇMİŞ 99

SİMGE LİSTESİ

L	Öneri fonksiyonu
L (u, y)	Ürün kullanıcı önerim fonksiyonu
M	Önerim yapılabilen alt kullanıcı grubu
S	Tüm ürün kümesini temsil eden fonksiyon

KISALTMA LİSTESİ

CF	Collaborative Filtering
CBR	Case Based Reasoning
JPA	Java Persistence API
OWL	Web Ontology Language
RDF	Resource Description Framework
RDF-S	Resource Description Framework Schema
RuleML	Rule Markup Language
SWL	Short Web Language
WWW	World Wide Web

ŞEKİL LİSTESİ

Sayfa

Şekil 3.1 Durum tabanlı çıkarsama grafiği 11
Şekil 4.1 Oluşturulan hibrit tavsiye sistemi 22
Şekil 4.2 Durum veritabanı ve ontolojiye dayalı oluşturulmuş RDF üçlüleri 26
Şekil 4.3 Bir CBR durumunun Environment sınıfının alt sınıflarının gösterimi 27
Şekil 4.4 CBR durumunun profil sınıfının ve alt sınıflarının gösterimi 28
Şekil 4.5 CBR durumun Preference sınıfının ve alt bileşenlerinin gösterimi 30
Şekil 4.6 Query sınıfın ontolojik gösterimi 31
Şekil 4.7 CBR durumunun sonuç ve geri bildirim bileşenlerinin gösterimi 31
Şekil 4.8 Çözüm sınıfını gösteren ontoloji modeli 32
Şekil 4.9 Çıkarsama harici CBR motorundan elde edilen önerinin gösterimi 33
Şekil 4.10 Çıkarsama ile birlikte CBR motorundan elde edilen önerinin gösterimi 34
Şekil 5.1 Sistem mimarisi 38
Şekil 6.1 Veri tabanı şeması 44
Şekil 6.2 Öneri sistemi web uygulaması giriş ekranı 46
Şekil 6.3 Öneri sistemi web uygulaması tavsiye sayfası 47
Şekil 7.1 Kullanıcı - Etiket verisi (%10 eğitim - 10 öneri) precision grafiği 51
Şekil 7.2 Kullanıcı - Etiket verisi (%10 eğitim - 10 öneri) recall grafiği 52
Şekil 7.3 Kullanıcı - Etiket verisi (%10 eğitim - 10 öneri) f-measure grafiği 52
Şekil 7.4 Kullanıcı - Etiket verisi (%50 eğitim - 10 öneri) precision grafiği 53
Şekil 7.5 Kullanıcı - Etiket verisi (%50 eğitim - 10 öneri) recall grafiği 53
Şekil 7.6 Kullanıcı - Etiket verisi (%50 eğitim - 10 öneri) f-measure grafiği 54
Şekil 7.7 Kullanıcı - Etiket verisi (%90 eğitim - 10 öneri) precision grafiği 55
Şekil 7.8 Kullanıcı - Etiket verisi (%90 eğitim - 10 öneri) recall grafiği 55
Şekil 7.9 Kullanıcı - Etiket verisi (%90 eğitim - 10 öneri) f-measure grafiği 56
Şekil 7.10 Kullanıcı - Etiket verisi (%10 eğitim - 20 öneri) precision grafiği 57
Şekil 7.11 Kullanıcı - Etiket verisi (%10 eğitim - 20 öneri) recall grafiği 57
Şekil 7.12 Kullanıcı - Etiket verisi (%10 eğitim - 20 öneri) f-measure grafiği 58
Şekil 7.13 Kullanıcı - Etiket verisi (%50 eğitim - 20 öneri) precision grafiği 59
Şekil 7.14 Kullanıcı - Etiket verisi (%50 eğitim - 20 öneri) recall grafiği 59
Şekil 7.15 Kullanıcı - Etiket verisi (%50 eğitim - 20 öneri) f-measure grafiği 60
Şekil 7.16 Kullanıcı - Etiket verisi (%90 eğitim - 20 öneri) precision grafiği 61
Şekil 7.17 Kullanıcı - Etiket verisi (%90 eğitim - 20 öneri) recall grafiği 61
Şekil 7.18 Kullanıcı - Etiket verisi (%90 eğitim - 20 öneri) f-measure grafiği 62
Şekil 7.19 Kullanıcı - Şarkı verisi tek ülke (%10 eğitim - 10 öneri) precision grafiği 63

Şekil 7.20 Kullanıcı - Şarkı verisi tek ülke (%10 eğitim - 10 öneri) recall grafiği 63
Şekil 7.21 Kullanıcı - Şarkı verisi tek ülke (%10 eğitim - 10 öneri) f-measure grafiği 64
Şekil 7.22 Kullanıcı - Şarkı verisi tek ülke (%50 eğitim - 10 öneri) precision grafiği 65
Şekil 7.23 Kullanıcı - Şarkı verisi tek ülke (%50 eğitim - 10 öneri) recall grafiği 65
Şekil 7.24 Kullanıcı - Şarkı verisi tek ülke (%50 eğitim - 10 öneri) f-measure grafiği 66
Şekil 7.25 Kullanıcı - Şarkı verisi tek ülke (%90 eğitim - 10 öneri) precision grafiği 67
Şekil 7.26 Kullanıcı - Şarkı verisi tek ülke (%90 eğitim - 10 öneri) recall grafiği 67
Şekil 7.27 Kullanıcı - Şarkı verisi tek ülke (%90 eğitim - 10 öneri) f-measure grafiği 68
Şekil 7.28 Kullanıcı - Şarkı verisi tek ülke (%10 eğitim - 20 öneri) precision grafiği 69
Şekil 7.29 Kullanıcı - Şarkı verisi tek ülke (%10 eğitim - 20 öneri) recall grafiği 69
Şekil 7.30 Kullanıcı - Şarkı verisi tek ülke (%10 eğitim - 20 öneri) f-measure grafiği 70
Şekil 7.31 Kullanıcı - Şarkı verisi tek ülke (%50 eğitim - 20 öneri) precision grafiği 71
Şekil 7.32 Kullanıcı - Şarkı verisi tek ülke (%50 eğitim - 20 öneri) recall grafiği 71
Şekil 7.33 Kullanıcı - Şarkı verisi tek ülke (%50 eğitim - 20 öneri) f-measure grafiği 72
Şekil 7.34 Kullanıcı - Şarkı verisi tek ülke (%90 eğitim - 20 öneri) precision grafiği 73
Şekil 7.35 Kullanıcı - Şarkı verisi tek ülke (%90 eğitim - 20 öneri) recall grafiği 73
Şekil 7.36 Kullanıcı - Şarkı verisi tek ülke (%90 eğitim - 20 öneri) f-measure grafiği 74
Şekil 7.37 Kullanıcı - Şarkı verisi (%10 eğitim - 10 öneri) precision grafiği 75
Şekil 7.38 Kullanıcı - Şarkı verisi (%10 eğitim - 10 öneri) recall grafiği 75
Şekil 7.39 Kullanıcı - Şarkı verisi (%10 eğitim - 10 öneri) f-measure grafiği 76
Şekil 7.40 Kullanıcı - Şarkı verisi (%50 eğitim - 10 öneri) precision grafiği 77
Şekil 7.41 Kullanıcı - Şarkı verisi (%50 eğitim - 10 öneri) recall grafiği 77
Şekil 7.42 Kullanıcı - Şarkı verisi (%50 eğitim - 10 öneri) f-measure grafiği 78
Şekil 7.43 Kullanıcı - Şarkı verisi (%90 eğitim - 10 öneri) precision grafiği 79
Şekil 7.44 Kullanıcı - Şarkı verisi (%90 eğitim - 10 öneri) recall grafiği 79
Şekil 7.45 Kullanıcı - Şarkı verisi (%90 eğitim - 10 öneri) f-measure grafiği 80
Şekil 7.46 Kullanıcı - Şarkı verisi (%10 eğitim - 20 öneri) precision grafiği 81
Şekil 7.47 Kullanıcı - Şarkı verisi (%10 eğitim - 20 öneri) recall grafiği 81
Şekil 7.48 Kullanıcı - Şarkı verisi (%10 eğitim - 20 öneri) f-measure grafiği 82
Şekil 7.49 Kullanıcı - Şarkı verisi (%50 eğitim - 20 öneri) precision grafiği 83
Şekil 7.50 Kullanıcı - Şarkı verisi (%50 eğitim - 20 öneri) recall grafiği 83
Şekil 7.51 Kullanıcı - Şarkı verisi (%50 eğitim - 20 öneri) f-measure grafiği 84
Şekil 7.52 Kullanıcı - Şarkı verisi (%90 eğitim - 20 öneri) precision grafiği 85
Şekil 7.53 Kullanıcı - Şarkı verisi (%90 eğitim - 20 öneri) recall grafiği 85
Şekil 7.54 Kullanıcı - Şarkı verisi (%90 eğitim - 20 öneri) f-measure grafiği 86

ÇİZELGE LİSTESİ

Sayfa

Çizelge 3.1 Örnek Kullanıcı – Ürün tablosu 8
Çizelge 4.1 Lastfm Veri İstatistikleri 19
Çizelge 4.2 Hibrit yaklaşım algoritması 37
Çizelge 6.1 Ham müzik verisinin dosya içindeki yapısı 41
Çizelge 6.2 Ham kullanıcı verisinin dosya içindeki yapısı 41
Çizelge 6.3 Ham müzik verisinin veri tabanına aktarılması 42
Çizelge 6.4 Track tablosu oluşturma 42
Çizelge 6.5 Ham kullanıcı verisinin veri tabanına aktarılması 43
Çizelge 6.6 Kullanıcı – Şarkı tablosunun oluşturulması 43
Çizelge 6.7 Örnek bir nesne gösterimi 45
Çizelge 6.8 Örnek CF girdisi 47
Çizelge 6.9 Örnek bir durum gösterimi 48
Çizelge A.1 Müzik ontolojisi için geliştirilen owl dosyası 93

ONTOLOJİ VE DURUMSAL TABANLI ÇIKARSAMAYA DAYALI İŞBİRLİKÇİ TAVSİYE YÖNTEMİ

Zafer DÜZEN

Bilgisayar Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Tez Danışmanı: Yrd. Doç. Dr. Mehmet S. AKTAŞ

İşbirlikçi filtreleme (CF), çok sayıda kullanıcının beğenilerini toplayarak, bir kullanıcının ilgilenebilecekleri hakkında tahminlerde bulunmaya dayalı bir tavsiye yöntemidir. İşbirlikçi tekniğinin kullanımında karşılaşılan en önemli problemlerden bazıları veri seyrekliğin dolayı ortaya çıkan soğuk başlangıç (cold-start) problemi oluşması ve kişiye özel öneriler yapılamamasıdır. Bunun sonucu olarak CF tabanlı tavsiye sistemlerinin ürettiği önerilerin başarım oranlarının arttırılması aktif araştırma alanları arasında yer almaktadır. Bu araştırma kapsamında önerdiğimiz yöntem, CF tekniğindeki bu eksikliklerin giderilmesi ve tavsiye sisteminin başarım oranının arttırılması için, Durumsal Tabanlı Çıkarsama (CBR) ve ontolojik veri zenginleştirme yöntemi kullanılarak hibrit bir yaklaşım yöntemi sunmaktadır. Geliştirdiğimiz hibrit tavsiye sisteminin kullanılabilirliğini göstermek adına önerilen yöntemi kullanan bir müzik dinleme asistanı geliştirilmiştir.

Anahtar Kelimeler: Durum Tabanlı Çıkarsama, İşbirlikçi Filtreleme, Hibrit Tavsiye Sistemleri, Ontolojik Veri Zenginleştirme

ABSTRACT

COLLABORATIVE RECOMMENDATION METHOD BASED ON THE ONTOLOGY AND CASE REASONING

Zafer DÜZEN

Department of Computer Engineering

MSc. Thesis

Adviser: Assoc. Prof. Dr. Mehmet S. AKTAŞ

Collaborating Filtering (CF) is a recommendation method that can make predictions about a given user's interest by collecting a large number of other user' appreciation. Some of the major problems encountered in the use of CF are the cold-start problem and the fact that personalized recommendations cannot be done. In turn, CF-based recommendations produces ranked results where the success rate can be improved. The method proposed in this research is a hybrid recommender system that utilizes Case-Based Reasoning (CBR) and ontological data enrichment in order to overcome these shortcomings and improve the success rate of the recommender system. To show the usability of the proposed hybrid recommender method, we have used a music recommendation dataset and build music listening assistant that uses the implementation of the method. The performance of the proposed method was evaluated and results are reported.

Keywords: Case-Based Reasoning, Collaborative Filtering, Recommender Systems, Hybrid Recommender Systems, Ontological Data Enrichment

BÖLÜM 1

GİRİŞ

1.1 Literatür Özeti

Tavsiye Sistemleri, günümüzde büyük verilerin olduğu sistemlerde oldukça yaygın bir şekilde kullanılmakta olup, kullanıcıların bu yüksek hacimli veri içinden daha kolay seçim yapabilmesini sağlayan sistemlerdir. Bu tip sistemler gün geçtikçe daha fazla alanda kullanılmakta olup, yaygın olarak kitap, müzik, haber, film, araştırma makaleleri ve e-ticaret ürünleri üzerinde yoğunlaşmaktadır. Öneri sistemlerinin çalışma prensibi genellikle kullanıcı tercihleri ve kullanıcıların ürünler üzerinde yaptığı puanlama veya yorumlamalara göre çalışmasıdır. Bu tarz tavsiye motorlarının artmasıyla birlikte, öneri sistemlerini gerçekleştiren algoritmalardan kaynaklı sorunların çözülmesi veya optimize edilmesi de akademik çalışmalarla yanıtlanmaya çalışılan araştırma soruları arasındadır.

1.2 Tezin Amacı

Bu tez kapsamında özellikle tavsiye sistemlerinde yaygın olarak kullanılan işbirlikçi filtreleme yöntemi üzerinde durulmaktadır. İşbirlikçi filtreleme, birçok kullanıcının tercihlerini, beğenilerini toplayarak, bir kullanıcının ilgilenebilecekleri hakkında tahminlerde bulunmaya dayalı bir tavsiye yöntemidir. İşbirlikçi filtreleme yönteminin örneklerinin, kullanıcı-tabanlı (User-Based Collaborative Filtering) ve ürün-tabanlı (Item-Based Collaborative Filtering) olarak iki ana kategori altında toplandığı görülmektedir. Birinci kategoride, beğenileri tahmin edilen kullanıcı ile aynı beğeni desenine sahip olan kullanıcılara bakılarak, kullanıcının ilgilenebilecekleri hakkında tahminler üreten örnekler yer almaktadır. İkinci kategoride, ürünler arasında bir ilişki

matrisi kurarak tahminler yapan işbirlikçi filtreleme örnekleri yer almaktadır. Böylesi bir ilişki matrisi üzerinde, bir X ürününü alan kullanıcıların bir başka Y ürününü almış olması gibi durumlar gösterilebilmektedir. Bu durumlar tespit edildiğinden, beğenileri tahmin edilecek olan kullanıcıya, aralarında ilişki tespit edilmiş ürünlerden birini almış olması durumunda diğer ürünler önerilmektedir. Tezimizde işbirlikçi filtrelemede sadece kullanıcı temelli işbirlikçi filtrelemeye dayalı önerilerin yapılması üzerinde yoğunlaşılmaktadır.

1.3 Hipotez

İşbirlikçi filtreleme yönteminde karşılaşılan en önemli problemin, beğenileri tahmin edilecek kullanıcı ile ilgili herhangi bir verinin söz konusu olmadığı durumda ortaya çıktığı görülmektedir. Bu problem yaygın olarak soğuk-başlangıç (cold-start) problemi olarak da literatürde yer almaktadır. Kullanıcının beğenilerine ait geçmiş veriler olmadığı durumda, aynı beğeni desenine sahip olan kullanıcılar tespit edilememekte ve doğru öneriler yapılamamaktadır. Bu tez kapsamında işbirlikçi filtreleme yöntemindeki bu probleme bir çözüm yaklaşımı sunulmaktadır. Bu yaklaşım, ontoloji ve durumsal çıkarsama tabanlı hibrit bir işbirlikçi filtreleme yöntemi içermektedir.

Tez kapsamında incelediğimiz araştırma probleminin tanımını ve bu problem çerçevesinde yanıtladığımız araştırma sorularını 2. Bölüm' de vermekteyiz. 3. Bölüm tez kapsamında izlediğimiz yöntemlerle ilgili temel bilgiler ve literatür taramasını anlatmaktadır. 4. Bölümde, önerdiğimiz metodolojiyi tartışıyoruz. 5. Bölüm önerdiğimiz metodolojiyi gerçeklemek için ortaya koyduğumuz yazılım mimarisini ortaya koymaktadır. 6. Bölüm önerdiğimiz yöntemin kullanılabilirliğini göstermek adına geliştirdiğimiz uygulamanın detaylarını açıklamaktadır. 7. Bölüm, önerilen yaklaşımın bir değerlendirmesini ortaya koymaktadır. 8. Bölümde değerlendirme sonuçlarını ve belirlediğimiz araştırma sorularına verdiğimiz yanıtları ve gelecek için planlanan araştırmaları tartışıyoruz.

BÖLÜM 2

ARAŞTIRMA PROBLEMİ

2.1 Problemin Tanımı

Tavsiye sistemleri oluşturmak için birçok yöntem geliştirilmiştir. Bu yöntemler genellikle üç ana kategoride toplanmaktadır. Bunlar işbirlikçi filtreleme teknikleri, içerik temelli teknikler ve bilgi tabanlı teknikler olmak üzere gruplandırılabilir [1].

İşbirlikçi filtreleme, farklı kullanıcıların geçmişteki seçimlerinden sonuç çıkarma yöntemine verilen isimdir. İçerik tabanlı filtrelemede ise kullanıcıların daha önceki seçimlerine bakarak tavsiye yapılan yöntemdir. Bilgi tabanlı filtrelemede ise kullanıcıların sistemde var olan bilgileri kullanılarak önermeler yapılır. Önerim modellemelerinin çözmesi gereken bazı temel zorluklar beş ana grupta toplanabilir. Bunlar veri eksikliği, değişen veri, kullanıcı tercihlerindeki değişiklikler, karmaşık veri yapıları ve tahmin edilemeyen veriler.

Bu tez kapsamında, önerim modeli olarak işbirlikçi filtreleme yaklaşımı benimsenmiştir. İşbirlikçi filtreleme önerim yaklaşımda, bu zorluklar arasında yer alan veri seyrekliği en önemli problem olarak kaşımıza çıkmaktadır. Veri seyrekliği kullanıcıların ilgilendiği ürünleri değerlendirmemesi veya az sayıda değerlendirmesi sonucu ortaya çıkan bir durumdur. Bu problemin sonucu olarak tavsiye motorunun ürettiği sonuçlar başarım olarak düşük seviyelidir [2]. Soğuk başlangıç problemi, işbirlikçi filtreleme yönteminde karşılaşılan ve beğenilerinin tahmini yapılan kullanıcı ile ilgili veri eksiliğinin olduğu durumda ortaya çıkan bir problemidir. Var olan bir sisteme eklenen yeni bir ürün veya sisteme yeni katılan bir kullanıcı olması durumunda tavsiye motorunun ürünü kimseye

öneremememesi durumu veya yeni katılan kullanıcıya önerilebilecek bir ürün bulamaması durumlarını tarif etmek için kullanılmaktadır [3]. Bu problemin çözümüne yönelik bazı yaklaşımlar, yeni katılan kullanıcıdan ek bilgiler alma veya eklenen yeni ürün için fazladan bilgi girilmesi zorunluluklarının getirilmesi ile çözülmektedir [4]. Literatürde yer alan geleneksel kullanıcı tabanlı veya ürün tabanlı işbirlikçi filtreleme yöntemlerinin kullanıcılarının demografik bilgilerinin ve karakter özelliklerinin birbirine benzerliğine değil sadece aldıkları ürünlerin benzerliğine bakarak karar verdikleri görülmektedir. Bunun da, işbirlikçi filtrelemeye yönelik tavsiye sistemlerinde başarımı düşüren sorunlardan biri olduğu görülmektedir [5]. İşbirlikçi öneri sistemlerinde önerilerin, çevrimdışı ve çevrimiçi olarak veriler üzerinde yapılan hesaplamalar sonucunda yapıldığı görülmektedir. Bu sistemlerde karşılaşılan diğer bir problem ise çevrimdışı yapılan algoritma hesaplamalarında kullanıcının tercih değişimlerin dikkate alınmaması ve daima eski tercihler üzerinden önerilerin yapılıyor olmasıdır [6]. Öneri sistemlerinin kararsız kullanıcılarla ilgili başarılı öneriler yapamadığı da görülmektedir. Kararsız kullanıcılar daimi olarak herhangi bir kişi veya grupla eşleştirilemeyen veya tavsiye sisteminden yararlanmak istemeyen kullanıcı gruplarıdır. Bu tip kullanıcıları, farklı öneri modelleri kullanan tavsiye motorları çözümleyemez ve ürün teklifinde bulunamamaktadır [7].

Günümüzde öneri sistemlerindeki kullanıcı ve ürün sayısı çok fazla olabilmektedir. Bu tip sistemlerde popüler ürünler çok az kullanıcının seçimleriyle ortaya çıkabilmektedir. Bu gibi durumlarda kullanıcılar arasındaki benzerlikler bulunamadığı görülmektedir. Yine, beğenileri tahmin edilecek kullanıcı ile ilgili herhangi bir geçmiş verinin söz konusu olmadığı durumda (cold-start problemi) da kullanıcılar arasında benzerlikler kurulamamaktadır. Soğuk başlangıç problemi ile karşılaşıldığı durumlarda sisteme yeni giren kullanıcıya öneri yapılamamaktadır veya sisteme yeni girilen ürün önerilerde kullanılamamaktadır. Geleneksel işbirlikçi filtreleme tekniklerin veri eksikliğinden ortaya çıkan bu tip durumlarda, doğru öneriler yapamadığı tespit edilmiştir [8]. **Bu tez, işbirlikçi filtrelemede karşılaşılan veri eksikliği sonucu doğru öneriler yapılamaması problemine bir çözüm sunmaktadır.** Bu problem tanımı ışığında, belirlediğimiz araştırma soruları maddeler halinde listelenmektedir.

Tavsiye sistemlerinde kullanılan işbirlikçi filtreleme yönteminde karşılaşılan soğuk başlangıç problemi nasıl çözülebilir?

Tavsiye sistemlerinde kullanılan işbirlikçi filtreleme yöntemi kullanıcılara ürün tavsiyesi yaparken karşılaştığı en önemli problem soğuk-başlangıç problemidir. Öneri sistemlerindeki bu problem sisteme katılan kullanıcılarla ilgili az bilgiye sahip olunması ve sonuç çıkarma işleminin yapılamaması durumunda karşılaşılmaktadır [3]. Bu tez kapsamında bu probleme çözmeye yönelik bir yaklaşım araştırılacaktır.

Ontoloji destekli Durum Tabanlı Çıkarsama Yöntemi ile farklılaşabilen kişisel profillere dayalı olarak öneriler yapabilen bir tavsiye sistemi nasıl geliştirilebilir?

Tavsiye sistemlerinde karşılaşılan problemlerden bir diğeri ise kullanıcı demografik bilgilerindeki farklılaşma ve kullanıcı tercihlerindeki değişikliklerdir. İşbirlikçi yöntemler kullanılırken sistemdeki kullanıcılar üzerinde gruplandırma işlemi yapılmaktadır. Bu gruplandırma işlemi kullanıcıların daha önceki seçimlerine bakılarak yapılmakta olup, kullanıcı tercihlerindeki zamanla oluşan değişiklikler fark edilememektedir [9]. Bu tez kapsamında ontoloji destekli durumsal çıkarsama yöntemi ile farklılaşabilen kişisel profillere dayalı olarak öneriler yapabilen bir tavsiye sistemi geliştirilmeye çalışılarak yukarıda değinilen probleme çözüm aranacaktır.

İşbirlikçi filtreleme ve Ontoloji Destekli Durumsal Çıkarsama Yönteminin birlikte kullanılması sonucu hibrit ve jenerik bir tavsiye sistemi için yazılım mimarisi nedir ve bu mimari yapı nasıl geliştirilmelidir?

Bu tez kapsamında, ontoloji ve durum tabanlı çıkarsama yöntemleri kullanılarak, kullanıcıların sistemdeki geçmiş deneyimlerinden yararlanarak yeni kullanıcılara öneri yapan işbirlikçi filtreleme yönteminin, nasıl geliştirileceği üzerinden çalışılacaktır. Bu kapsamda, işbirlikçi filtreleme ve ontoloji destekli durumsal çıkarsama yönteminin birlikte kullanılması sonucu hibrit ve jenerik bir tavsiye sistemi için uygun yazılım mimarisinin ne olduğu ve nasıl geliştirileceği soruları da yanıtlanacaktır. İşbirlikçi filtreleme yönteminin eksikliklerine çözüm sunan ve kişiye özgü ve yüksek başarımlı öneriler yapabilen bir tavsiye sisteminin nasıl oluşturulacağı aktarılacaktır. Başarımı yükseltmeyi hedefleyen bu modellemede kullanılan teknolojiler üzerinde durulacaktır.

Müzik alanında kişilerin ilgilerini modelleyen bir ontoloji nasıl geliştirilir, nasıl olmalıdır?

Geliştirilen sistemin genellik içermesi hedefler arasındadır. Ancak sistemin kullanılabilir olduğunun gösterilmesi adına, bir uygulama alanı seçilmiştir. Bu sebeple, Müzik, uygulama alanı olarak belirlenmiştir ve lastFM müzk veri seti kullanılmıştır. Müzik alanında yer alan bilgi varlıkları ve bilgi varlıkları arasındaki ilişkiler incelenerek, bir müzik ontolojisinin nasıl oluşturulacağı problemi, bu tez kapsamında araştırılacaktır. Geliştirilecek olan Müzik ontolojisiyle, işbirlikçi filtreleme ve durum tabanlı çıkarsama yöntemleri kullanılarak yeni kullanıcılara öneriler yapabilen bir sistem geliştirilmesi sağlanacaktır.

Klasik İşbirlikçi Filtrelemeye Dayalı Tavsiye Yöntemi, Durumsal Çıkarsama Yöntemine dayalı Tavsiye Sistemi ve bu araştırmada önerilen Hibrit Tavsiye sistemi yöntemlerinin doğrulukları nasıl karşılaştırılmalıdır ve sonuçlar nasıl değerlendirilmelidir?

Bu tez kapsamında klasik işbirlikçi filtrelemeye dayalı, durumsal çıkarsama yöntemine dayalı ve önerdiğimiz bu iki yöntemin ortak kullanımı sonucu oluşan hibrit yönteme dayalı tavsiye sistemlerinin doğruluklarının karşılaştırmalı olarak nasıl değerlendirileceği çalışılacaktır. Sonuçlar değerlendirilerek önerilen çözümün kullanılabilirliği tartışılacaktır.

BÖLÜM 3

TEMEL BİLGİLER ve LİTERATÜR ÇALIŞMASI

3.1 Temel Bilgiler

Günümüzde kullanılan tavsiye sistemlerinde farklı yöntemler kullanılabilmektedir. Bu tez kapsamında geliştirdiğimiz tavsiye sisteminde önerdiğimiz hibrit yöntemin daha iyi anlaşılabilmesi için, önerdiğimiz yöntemin alt bileşenlerini oluşturan yöntemleri - İşbirlikçi Filtreleme (Collaborative Filtering), Durum tabanlı çıkarsama (Case Based Reasoning) ve Ontolojiler – bu bölümde detaylı olarak açıklıyoruz.

3.2 İşbirlikçi Filtreleme Metodolojisi

İşbirlikçi filtrelemenin amacı kullanıcı tercihlerinin veya davranışlarının modellemelerini çıkararak veriden öneri modelleri elde etmektir [10]. Bu yüzden işbirlikçi filtreleme bir kullanıcının geçmişte yaptığı seçimlerden yola çıkarak bu kullanıcının tercih modellerini oluşturulmasını ve kullanıcıya yeni öneriler sunabilmeyi sağlamaktadır. Geçmişte aynı ürün üzerinde karar kılan kullanıcılar gelecekte de aynı ürünü seçmeye eğilimlidirler. Bu varsayım işbirlikçi yöntemlerin temelini oluşturmaktadır.

İşbirlikçi filtreleme ilk olarak *Xerox* araştırma merkezinde e-posta filtreleme amacıyla 1990lı yıllarda kullanılmıştır. Bu amaçla oluşturulan uygulama küçük bir kullanıcı grubu tarafından kullanılmış ve başarı sağlanmıştır. İşbirlikçi filtreleme genel olarak kitap, film, müzik, e-ticaret ürünleri vb. önerim sistemlerinde sıklıkla kullanılmaktadır [11].

Öneri sistemlerinin amacı bir kullanıcı için yapılan önerinin doğruluğunu maksimuma getirebilmektir. Örnek olarak bir sistemdeki tüm kullanıcılar U tüm ürünler ise Y olarak seçilmiş olsun. L bir öneri fonksiyonu iken, seçilen bir $u \in U$ kullanıcısı için ve bu kullanıcının seçmiş olduğu $y \in Y$ ürünü için;

$$L : U \times Y \rightarrow S \tag{3.1}$$

(3.1) eşitliğindeki S, tüm ürünlerin alt kümesi olup u kullanıcısı için önerileri içeren denklemdir. İşbirlikçi filtrelemede ise denklemin içine diğer kullanıcıların seçimleri de katılmaktadır.

$$L(u_1, y), L(u_2, y), \ldots . L(u_m, y) \; : \; L(u, y), \qquad u_i \in M(u) \tag{3.2}$$

L(u,y) diğer kullanıcıların fonksiyonlarını göstermektedir. Bu işbirlikçi filtrelemenin, işbirliği kısmını oluşturan fonksiyondur. Denklemdeki M ise filtreleme bölümünü göstermektedir. M fonksiyonunun görevi veri seti içerinden yüksek kalitede alt kullanıcı grupları seçmektir. Çizelge 3.1, işbirlikçi filtrelemenin çalıştığı kullanıcı-ürün matrisini göstermektedir. Bu matris kullanıcıların belli ürünlere verdiği puanları temsil etmektedir. Kullanıcı 4'e sistemden öneri yapılacağı zaman diğer kullanıcılarla, Kullanıcı 4'ün ürünlere verdiği puanlar karşılaştırılır ve Kullanıcı 4'e en çok benzer olandan başlayarak Ürün 3 için öneriler yapılır. Bu matriste benzerlik "kullanıcı-kullanıcı" tipinde veya "kullanıcı-ürün" tipinde olarak hesaplanabilmektedir.

Çizelge 3.1 Örnek Kullanıcı – Ürün tablosu

	Ürün 1	***Ürün 2***	***Ürün 3***	***Ürün 4***
Kullanıcı 1	5	3	2	1
Kullanıcı 2	3	4	2	1
Kullanıcı 3	4	4	4	3
Kullanıcı 4	4	5	?	1

İşbirlikçi filtreleme algoritmaları genel olarak iki sınıfta gruplandırılabilirler. Bunlar Hafıza tabanlı sistemler ve model tabanlı sistemler olarak adlandırılmaktadırlar. Hafıza tabanlı sistemler öneri yapabilmek için tüm kullanıcı verisi üzerinde işlem yaparlar. Bu tip sistemler istatistiki verilere dayanmaktadırlar. Model tabanlı sistemler ise ilk olarak kullanıcı verisini kullanarak bir filtreleme modeli oluşturur ve daha sonra önerilerini bu model üzerinden yapar. Model oluşturma işlemi için SVD, PCA, Bayesian network ve kümeleme makine öğrenmesi metotları kullanılmaktadır. Ayrıca bu tipte olan işbirlikçi yöntemler çevrimiçi ve çevrimdışı olarak da tavsiyeler sunabilmektedir [12]. Bu tez kapsamında işbirlikçi filtreleme yöntemi kullanılmaktadır.

3.3 Durum Tabanlı Çıkarsama Metodolojisi

Durum tabanlı çıkarsama geçmiş deneyimlerden yararlanılarak yeni problemlere çözüm bulmak için geliştirilmiş bir karar verme yöntemidir. Olay tabanlı çıkarsama olarak da adlandırılan bu yöntem bir problemin çözümünü geçmişte karşılaşılmış ve çözülmüş olan problemlerle karşılaştırır ve benzer problem çözümlerinin kullanarak seçim yapılmasını sağlar [13].

Durum tabanlı çıkarsama yapay zekâ algoritmalarından biri olup insanların problem çözme davranışlarının taklit edilmesini sağlamaktadır. Bu algoritmanın temelinde geçmiş problemleri hatırlama yatmaktadır. İnsanlar bir problemle karşılaştığında, ilk olarak hafızasında bu problemle daha önce karşılaşıp karşılaşmadığına bakar. Eğer bu problemle karşılaştıysa o zaman var olan çözümü kullanır. Eğer bu problemle karşılaşmadı ise de o zaman probleme bir çözüm bulunur ve bu çözüm belleğe geçmiş bir deneyim olarak kaydedilir. Olay tabanlı çıkarsama algoritmaları insanların bu davranış tipini taklit etmektedir [14].

Durum veya olay, karşılaşılan bir sorun veya problemin özelliklerine, niteliklerine ve diğer problemlerle arasındaki ilişkilere dayandırılarak oluşturulan bir problem tasarımıdır. Bir durumun yapısı bir çalışma alanından diğerine değişmektedir. Bu durumun veya olayın probleme özel olduğunu göstermektedir [15]. Olay tabanlı çıkarsamanın ilk örnekleri yapay zekâ alanında makine öğrenmesi yöntemleri ile kullanılmıştır. Bu çalışmaların birinde dinamik bellek sorunlarının, problem çözme ve çözümü öğrenmeye dayalı olarak çözülmesi amaçlanmıştır [16].

Kolodner'e [13] göre durum tabanlı çıkarmanın özellikleri aşağıdaki gibidir.

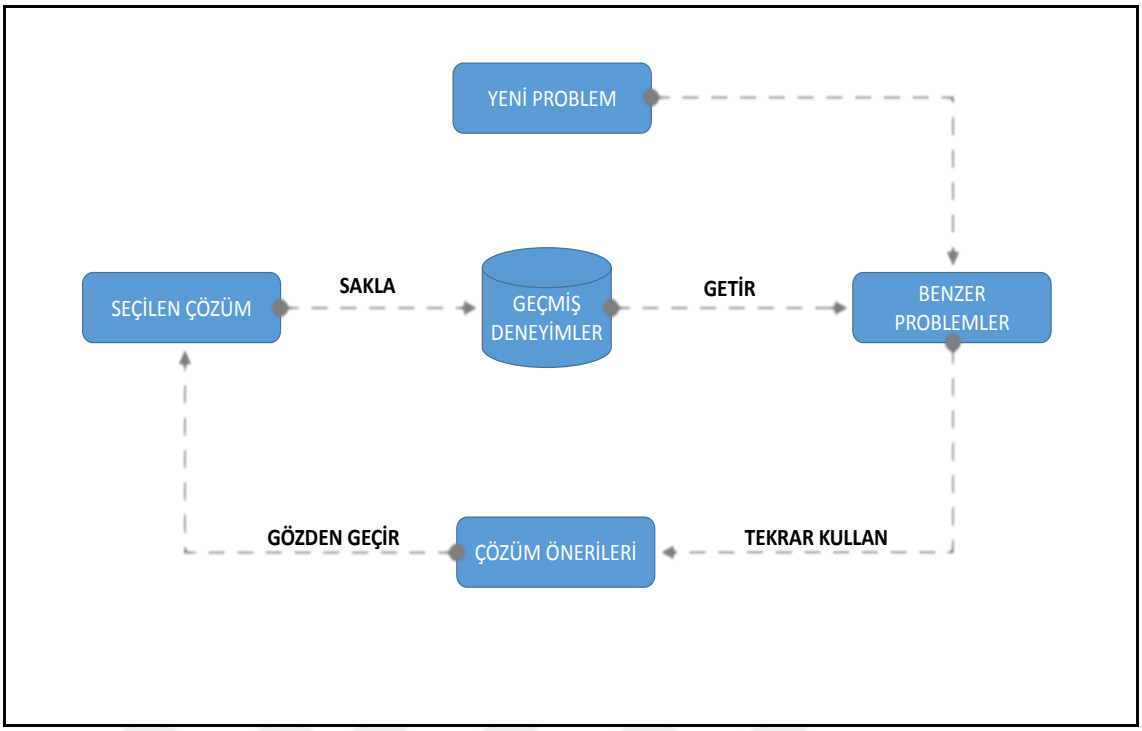
- Deneyimlerden öğrenme,
- Neden-sonuç ilişkisi kurabilme,
- Yapılan yanlışları öğrenme ve bunları tekrarlamama,
- Problemleri minimum çaba ile çözme,
- Geçmişte doğru olanın şimdi de doğru olacağını varsayma.

Hammand' a [17] göre ise iyi bir durum tabanlı çıkarsama aşağıda belirtilen özellikleri sağlamaktadır.

- Çalışıyorsa, tekrar kullan
- Çalışıyorsa, elinde çözüm var demektir.
- Çalışmıyorsa, tekrar kullanmaman gerektiğini hatırla
- Çalışmıyorsa, düzelt

Durum tabanlı çıkarsama algoritmaları dört ana adımdan oluşmaktadır. İlk adımda durumun veya olayın yapısal tasarımı oluşturulur. İkinci adımda var olan durumlar veri tabanından alınır. Üçüncü adım ise benzer durumların hesaplanması ve kullanıcıya sunulmasıdır. Son olarak da kabul edilen çözümün yeni bir durum olarak kaydedilmesi vardır. Aamand ve Plaza olay tabanlı çıkarsama sürecini döngüsel bir yapıda dört adımda tanımlamışlardır. Şekil 3.1 durum tabanlı çıkarsamanın adımlarını göstermektedir [18].

- Getir (Retrieve)
- Tekrar Kullan (Reuse)
- Gözden Geçir (Revise)
- Sakla (Retain)



Şekil 3.1 Durum tabanlı çıkarsama grafiği

Yeni bir problem ile karşılaşıldığında durum veri tabanından bu probleme benzer olan tüm problemler getirilir. Çözüm olarak benzer durumlardan önerim işlemi yapılır. Eğer önerilen durumlar içerisinde kullanıcıya uygun varsa ve kullanıcı bir seçim yaparsa tekrar kullan (reuse) adımı işletilir. Eğer önerilen durumlar kullanıcı için bir çözüm değilse, kullanıcının seçtiği çözüm üzerinden düzeltme (revise) işlemleri yapılır ve yeni bir durum olarak veri tabanına kaydedilir. Durumları elde etmek için "nearest neighbor", "induction" gibi algoritmalar kullanılmaktadır. Bu algoritmalar tek başına kullanılabildiği gibi birlikte kullanılarak hibrit yöntemlerde oluşturulabilir [19].

Durum tabanlı çıkarsamayı diğer makine öğrenmesi metotlarından ayıran en önemli özellik kendi limitlerini kendi belirleyebilmesidir. Eğer sistem benzer durum bulamaz ise tahmin de yapamaz. Bu şekilde öneri yapamama durumu iyi bir özellik gibi durmasa da diğer makine öğrenmesi metotları gibi anlamsız önerimler yapmaktan daha doğru bir harekettir. Ancak olay tabanlı çıkarsamanın da dezavantajları vardır. Bunlardan en önemlisi durum veri tabanındaki problem eksikliğidir. Sistemin yeterli derecede geçmiş deneyimi yoksa düzgün önerimler yapamadığı görülmektedir [20]. Bu tez kapsamında durum tabanlı çıkarsama yöntemi kullanılmaktadır.

3.4 Ontoloji

Ontolojiler belli bir alandaki bilgiyi formal bir altyapı kullanarak, bir dizi kavram ve bu kavramlar arasındaki ilişkiler yolu ile tanımlayan bilgi modelleridir. Anlamsal Web'in altyapısının son yıllarda hızla olgunlaşması, birçok alanda ontoloji geliştirme etkinliklerini arttırmaktadır. Tüm bu ontoloji geliştirme deneyimleri, ontoloji geliştirme sürecinde standart bir yöntemin kullanılmasının önemine işaret etmektedir. Bu tarz bir yöntemin uygulanması ile ontoloji geliştirme, birkaç uzmanın yaptıklarıyla sınırlı bir sanat olmaktan çıkarak; gereksinimlerin modellenmesinden doğrulanmasında kadar geçen sürecin etkinliklerinin ve çıktılarını tanımlandığı, belgeme yapıldığı, tekrarlanabilir ve bakımı yapılabilir bir mühendislik etkinliğine dönüşmektedir.

Anlamsal Web, ontoloji tanımlama dilleriyle webde dağıtık olarak sunulan içeriğin, insanların yanı sıra yazılımların da anlamlandırabilir olmasını amaçlamaktadır. Özellikle, webde yayınlanan içerikler arasındaki ilişkilerin de ontolojilerle tanımlanabilmesine olanak sağlayan bağlı veri yaklaşımı webin sorgulanabilir büyük bir veri tabanına dönüşmesini sağlamaktadır. Böylece her geçen gün daha da büyüyen Web'i insanlar gibi akıllı yazılımların da kullanmasının önü açılmaktadır.

Anlamsal Web ontolojilerinin temel yararı kurum içi ve kurum dışı bilgi bütünleştirilmesini kolaylaştırmasıdır. Bu bütünleştirme anlamsal webin temel yapı taşı olan ontoloji dilleri ile sağlanmaktadır. RDF, RDF-S, OWL gibi anlamsal web dilleri WWW Konsorsiyumu tarafından standartlaştırılmaktadır ve 2000'lerin başından beri süregelen bu standardizasyon süreci ile bu diller endüstride kullanılabilecek olgunluğa erişmişlerdir. Bu standart dilleri kullanarak geliştirilen ve farklı alanları modelleyen ontolojiler, diğer ontolojileri tekrar kullanabilmekte ve farklı ontolojilerdeki kaynaklara bağlanabilmektedirler. Bu bağlanabilme özelliğiyle dağıtık bilgi modelleri bütünleştirilmekte ve yazılımlar tarafından karar destek, bilgi sorgulama, öneri yapma gibi farklı amaçlar bağlamında kullanılabilmektedirler.

RDFS, OWL gibi anlamsal web ontoloji tanımlama dilleri, tanımlama mantığından ("Description Logic") gelen kökleri sayesinde "subclassOf", "transitive" ve "symmetric" gibi birçok özellik tanımıyla zengin çıkarsamalar yapılmasına olanak tanımaktadırlar. Öte yandan SWL ve RuleML gibi kural tanımlama dilleri sayesinde geliştiriciler

tarafından da yeni kurallar tanımlanması mümkün kılınmıştır. Alan ontolojilerinin tanımlanması sırasında bu diller aracılığıyla alana ait iş kurallarının da tanımlanması anlamsal web üzerinde çalışacak uygulamanın çıkarsama yeteneğini artırmakta, iş süreçlerinin daha etkin yönetilmesini, veri üretiminin ve denetiminin daha kolay yapılmasını sağlamaktadır. Bu tez kapsamında ontolojiler kullanılarak verinin tanımlanması sağlanmaktadır.

3.5 Literatür Çalışması

Günümüzde verilerin yoğunluğu ve sayısı üstel olarak artış göstermektedir. Bu artışa karşı bilgi arama ve karar verme sistemleri geliştirilmeye çalışılmaktadır. Bu sebeple akıllı ve kişiselleştirilmiş bilgi filtreleme sistemleri ortaya konulmaktadır. Tavsiye veya öneri sistemleri olarak adlandırılan uygulamalar bu tip akıllı sistemlerinin bir alt türünü oluşturmaktadırlar. Tavsiye sistemleri için birçok uygulama tekniği geliştirilmiştir. Bunlar işbirlikçi teknikler, içerik tabanlı teknikler ve bilgi tabanlı teknikler olmaz üzere üç ana grupta incelenebilirler [21].

3.6 İşbirlikçi Filtreleme Teknikleri Üzerindeki Kaynak Çalışması

İşbirlikçi filtreleme teknikleri ikiye ayrılmaktadır. Bunlar kullanıcı tabanlı ve parça tabanlı işbirlikçi filtreleme yöntemleridir. Bu yöntemlerde öneriler kullanıcıların diğer kullanıcılarla olan benzer seçimlerine veya puanlamalarına bakılarak yapılmaktadır [7]. İşbirlikçi filtreleme sistemleri veri yoğunluğunun fazla olduğu web sayfaları, film, müzik, kitap, makale, restoran vb. alanlarda öneri sistemlerinde yoğun olarak kullanılmaktadır. İşbirlikçi filtreleme yöntemleri e-ticaret sistemlerine de kolaylıkla uyum sağlayabilmektedirler. Örneğin amazon.com e-ticaret sitesi işbirlikçi filtreleme tekniklerinin ilk uygulayıcılarındandır [22]. İşbirlikçi tekniklerin tavsiye sistemlerinde yoğun olarak kullanılmaya başlamasıyla birlikte, buna benzer teknikerin bazı dezavantajları olduğu ortaya çıkmıştır. Karşılaşılan en önemli problemlerden biri veri seyrekliğidir. Veri seyrekliği kullanıcıların ilgilendiği ürünleri değerlendirmemesi veya az sayıda değerlendirmesi sonucu ortaya çıkan bir durumdur. Bunun problemin sonucu olarak tavsiye motorunun ürettiği sonuçlar başarım olarak düşük seviyelidir [2]. Soğuk-Başlangıç bir sistemde karşılaşılan ilk çalıştırma problemidir. Var olan bir sisteme

eklenen yeni bir ürün veya sisteme yeni katılan bir kullanıcı olması durumunda tavsiye motorunun ürünü kimseye önerememesi durumu veya yeni katılan kullanıcıya önerilebilecek bir ürün bulamaması durumudur [3]. Ayrıca bu tip sistemler kullanıcıların puanlamasına bağlıdır ve yüksek boyutlu verilerde ölçeklenebilirlik problemleri oluşmaktadır [7].

3.7 Bilgi Tabanlı Sistemler Üzerindeki Kaynak Çalışması

Bilgi tabanlı sistemler, tavsiye sistemlerinin diğer bir alt kolunu oluşturmaktadır. Bu tip sistemler işbirlikçi veya içerik tabanlı sistemlerinin kullanılamadığı alanlarda kullanılmaktadır. Bilgiye dayalı sistemlerin diğer öneri sistemlerinden farkı kullanıcılar ve ürünler hakkında bilgi toplaması ve ürettiği önerilerin kullanıcıların gereksinimi ile örtüşüp örtüşmediğini kontrol etmesidir [9]. Bilgiye dayalı sistemleri diğer öneri sistemlerinden ayıran en önemli avantajı ise soğuk-başlangıç problemine çözüm sunabilmesidir [9]. Bir restoran öneri sistemli olan “Entree” yapacağı tavsiyeleri, sistemdeki kullanıcıların daha önceden gitmiş veya beğenmiş olduğu restoranlara bakarak, diğerlerine benzer tavsiyeler sunabilmektedir [9]. Ancak bu tip sistemlerinde karşı karşıya olduğu problemler vardır. Bilgiye dayalı sistemlerin dezavantajları iki ana grupta toplanabilir. Bunlar bilgi çıkarımı ve önerilerin dinamik değil statik bir yapıda olmasıdır. Bilgi çıkarımı bu tip modellemelerdeki en zor kısımdır. Doğru öneriler yapabilmek için sistemdeki ürünlerin ve kullanıcıların derinlemesine bilgilerine ihtiyaç duyulmaktadır [23].

3.8 İçerik Tabanlı Filtreleme Sistemleri Üzerindeki Kaynak Çalışması

İçerik tabanlı filtreleme sistemleri önerilerini ürünün ait olduğu içerik ve kullanıcıların tercihlerine göre yapar. İşbirlikçi filtrelemede kullanıcılara benzer kullanıcılardan öneriler yapılırken bu tip sistemlerde kullanıcı tercihleri ön plana çıkmıştır [24]. Ayrıca içerik tabanlı sistemlerde kullanıcıların diğer kullanıcılardan bağımsızlığı ve soğuk-başlangıç sorununa karşı çözüm üretilmiştir [25]. İçerik tabanlı bilgi filtreleme sistemleri e-ticaret, sinema, kitap vb. alanlarda kullanılabilmektedir. LIBRA uygulaması Naive-Bayes metin kategorilendirme algoritmasını kullanarak web sayfalarından kitap önerileri yapabilmektedir [25]. Diğer filtreleme yöntemlerinde olduğu gibi bu tip

sistemlerinde olumsuz yanları vardır. Bu tip modellemelerde ancak limitli içerik analizi yapılabilmektedir. Yani doğruluk payı yüksek bir öneri yapabilmek için ürünün tüm bilgilerine ve kullanıcıların tüm tercihlerini bilmek gereklidir [25].

3.9 Hibrit Tavsiye Sistemleri Üzerindeki Kaynak Çalışması

Hibrit tavsiye sistemlerinde ise iki veya daha fazla teknik birlikte kullanılarak diğer sistemlerin zayıflıklarının önüne geçilmesine ve sistem performansının arttırılmasına çalışılmıştır. Bu şekilde yapılan bir modellemeyle tahminlerdeki doğruluk payının arttırılması hedeflenmiştir. Hibrit yöntemler veri setinin farklı kullanım durumları olduğunda veya farklı bakış açıları katılmak istenildiğinde kullanılmaktadır. Örneğin ürün tabanlı işbirlikçi filtreleme hiçbir kullanıcı ürünü puanlamadığında problem yaşarken, içerik bazlı filtrelemede bu durumla karşılaşılmamaktadır. Bir hibrit öneri sistemi ürün meta verisini kullanarak, tavsiye sistemlerinin gücünü arttırabilmektedir. Yapılan bir çalışmada ürünleri, kullanıcıları daha önce puanladığı içerik meta verileriyle eşleştirerek benzer bir öneri sistemi anlatılmaktadır [26].

Hibrit tavsiye sistemleri yedi temel grupta toplanmaktadırlar [9].

- Ağırlıklı hibrit tavsiye sistemleri gerekli oranları kullanılan tavsiye motorlarından alır ve bunları birleştirerek kullanıcıya iletir.
- Değişmeli hibrit tavsiye yöntemleri duruma bağlı olarak farklı algoritmaları kullanan tavsiye motorlarını seçme prensibine dayanır.
- İç içe geçmiş hibrit tavsiye sistemleri ağırlıklı tavsiye sistemlerine benzer fakat tek liste vermek zorunlu değildir.
- Özellik-birleştirici hibrit öneri sistemleri çoklu tavsiye veri setini bir arada kullanarak tek bir tavsiye algoritması kurmaktadır.
- Kademeli hibrit öneri sistemleri, birkaç öneri motorunun birbiri ardına çalışması prensibine dayanmaktadır.
- Arttırmalı hibrit öneri sistemleri bir algoritmanın çıktısının sonraki algoritmanın girdisi olması temeline dayanmaktadır.
- Meta-Seviyeli hibrit tavsiye sistemleri bir veri modelini algoritmalardan birini kullanarak veri yapısı öğrenimi için kullanır ve bu girdiyi diğer algoritmalara sunar.

Literatürde yer alan farklı hibrit yaklaşımlardan özellikle, durum tabanlı çıkarsama ve işbirlikçi filtreleme yöntemlerini birlikte kullanan çözümler üzerinde literatür taraması odaklanmıştır. Smyth ve diğerleri tarafından önerilen hibrit öneri yöntemi, kullanıcıların favori kanal seçimleri, ilgi alanları ve izleme zamanlarına göre işbirlikçi filtreleme ve durum tabanlı çıkarsama metotları birleştirerek bir öneri sistemi geliştirmiştir [27]. Chedrawy ve diğerleri tarafından önerilen bir diğer yaklaşımda, işbirlikçi filtreleme ve durum tabanlı çıkarsama metotları birleştirilerek kişiselleştirilmiş ürün önerileri yapılmaktadır. Bu yaklaşımın ilk fazında işbirlikçi filtreleme metodu kullanılarak öneriler elde edilmektedir. İkinci fazda ise durum tabanlı çıkarsama yönteminin adaptasyon adımı uygulanarak, yapılan başarılı önerilerinin birer durum olarak sisteme alınması ve saklanması işlemi yapılmaktadır [28]. Karim ve diğerleri tarafından önerilen bir başka yaklaşımda, yine işbirlikçi filtreleme ve durum tabanlı çıkarsama metodolojileri birlikte kullanılarak hibrit bir model üretilmiştir [21]. Burada e-turizm uygulaması ile kullanıcılara turistik olarak gidilebilecek mekân önerileri sunulmuştur. Yapılan uygulamada kullanıcıların geçmiş problemlerinden ve çözümlerinden oluşturulmuş bir durum veri tabanı ile tüm kullanıcıların geçmiş seçimlerini tutan bir veri tabanı kullanılmıştır. Bu iki veri kaynağı işbirlikçi filtreleme ve durum tabanlı çıkarsama öneri motorlarına verilerek tavsiyeler elde edilmiştir.

3.10 Hibrit Yaklaşımlardaki Eksiklikler

Durum tabanlı çıkarsama ve işbirlikçi filtreleme yöntemlerini birlikte kullanan çözümler incelendiğinde [27], [28], zamanla değişebilen kullanıcı görünüşlerinin takip edilebilmesine izin veren bir veri modellenmesine dayalı öneriler yapılamadığı görülmektedir. Bu yöntemler arasında, Karim ve arkadaşları tarafından önerilen hibrit yaklaşım [21] bu eksikliği gidermek ve geçmiş kullanıcıları deneyimlerini daha modellemek için ontoloji kullanımını önermektedir. Ancak bu yaklaşım da, ontoloji verileri üzerinde çıkarsama yaparak, kullanıcı bilgileri ya da deneyiminde olabilecek veri eksikliklerinin tamamlanmasına yönelik çözümler sunmamaktadır. Karim ve diğerlerinin yaklaşımı statik ve değişmeyen ontolojik verilerden oluşan bir kullanıcı veri kümesini öngörmektedir. Karim ve diğerleri tarafından yapılan bu öneri, gerçeklenmemiş ve önerilen yöntemin doğruluk değerlendirmesi ile ilgili bir veri de sunulmamıştır.

3.11 Tez Kapsamındaki Öneri

Bu tez kapsamında tavsiye sistemlerinde kullanılmakta olan CBR ve İşbirlikçi Filtreleme yöntemlerin başarımının arttırılması üzerinde araştırma yapılmaktadır. Bu yöntemler arasında yer alan Karim ve diğerleri tarafından önerilen tavsiye modelinde tespit ettiğimiz ve yukarıda belirttiğimiz eksiklikleri gideren, ontoloji tabanlı bir yaklaşım çözüm olarak belirlenmiştir. Önerdiğimiz çözümde işbirlikçi filtreleme metodu ile ontoloji destekli çıkarsama yöntemi birlikte kullanılarak farklılaşabilen kişisel profillere dayalı öneriler yapabilen sistemin nasıl geliştirileceği üstünde durulmaktadır. Bu kapsamda, kullanıcı profillerinin modellenmesi için bir ontoloji önerilmektedir ve uygulanacak hibrit öneri sistemi yaklaşımının nasıl olması gerektiği konusu ele alınmaktadır. Tez kapsamında geliştirilecek öneri sisteminin doğruluk derecesi, hibrit tavsiye modelinin temelini teşkil eden işbirlikçi filtreleme ve durum tabanlı çıkarsama yöntemleri ile karşılaştırmalı olarak incelenmektedir.

BÖLÜM 4

METODOLOJİ

Bu tez kapsamında geliştirdiğimiz yöntem, tavsiye sistemlerinde kullanılan iki tekniğin birlikte kullanılarak, bu tekniklerin zayıflıklarının önüne geçilmesi ve sistem performansının arttırılmasını sağlamaktadır. Bu yolla tavsiye sisteminin sağladığı tahminlerdeki doğruluk payının arttırılması hedeflenmektedir. Önceki bölümlerde açıkladığımız işbirlikçi filtreleme yöntemi ve durum tabanlı çıkarsama yöntemi birlikte kullanılarak, hibrit bir yöntem geliştirilmiştir. **Geliştirdiğimiz hibrit yöntemde, işbirlikçi filtreleme yönteminde karşılaşılan "soğuk başlangıç" problemine bir çözüm yaklaşımı sunulmaktadır.** Bu açıdan, çözüm olarak sunduğumuz öneri sistemi, sisteme yeni kayıt olan kullanıcılara daha önceden yaptıkları herhangi bir seçim olmadığı takdirde bile, anlamlı bir tavsiyede bulunulabilecek şekilde tasarlanmıştır. Önerdiğimiz çözüm, ontolojik veriler üzerinde çıkarsama yaparak, Bölüm 2' de detaylarını verdiğimiz veri eksikliği problemine de çözüm sağlamaktadır.

4.1 Uygulama Alanı

Önerdiğimiz yöntem öneri sisteminin kullanılacağı alanda bilgi varlıkları ve ilişkilerinin ontolojiler yardımıyla modellenmesini gerektirmektedir. Sistemin kullanılabilirliğini göstermek amacıyla, bir CBR Durum Ontolojisi geliştirilmiştir. Bu ontoloji için uygulama alanı olarak da Müzik alanı seçilmiştir. Bu seçim yapılırken, yöntemin yapacağı önerilerdeki doğruluk başarı oranlarının tespit edilmesi için kullanılacak veri setinin Müzik alanında olması etkin olmuştur. Metodolojimizin önemli bir parçası olan ontoloji tabanlı veri modellemesinden bahsedilmek için, müzik alanındaki CBR Durum

Ontolojisini geliştirirken, bilgi varlıklarını tespit etmek için kullandığımız veri setini aşağıda detaylı olarak veriyoruz.

4.2 Veri Seti

Üzerinde çalışılan veri LastFm isimli bir müzik sitesine aittir [29]. Kullanıcıların belli bir zaman dilimi üzerinde dinlediği müzikleri ve bunların dinleme tarihlerini tutmaktadır. Çizelge 4.1, LastFm verisi ile ilgili istatistiksel bilgileri sunmaktadır.

Çizelge 4.1 Lastfm veri istatistikleri

Veri setinin kaynağı	LastFm
Verinin alındığı zaman dilimi	2005-2019
Toplamdaki kullanıcı sayısı	992
Toplamdaki dinlenen şarkı (track) sayısı	19.150.868
Filtrelenmiş Tekil Dinleme Sayısı	9.421.151
Toplam Tekil Şarkı Sayısı	563.200
Toplam Tekil Etiket Sayısı	14.671
Kullanıcı-Etiket Tekil Dinleme Sayısı	41.794.338
Kullanıcı-Şarkı Gruplanmış Dinleme Sayısı	9.421.151
Kullanıcı-Etiket Gruplanmış Dinleme Sayısı	1.060.814

4.3 Önermekte Olunan Tavsiye Sistemindeki İş Akışı

Geliştirdiğimiz tavsiye sisteminin kullanımı, kullanıcının sisteme giriş yapmasıyla başlamaktadır. Sisteme ilk defa gelen kullanıcılardan, profil bilgileri (demografik bilgiler, öncelikler v.b.) istenmektedir. Sistemde kullanıcı hesabı olan kullanıcıların, sisteme kullanıcı adlarıyla giriş yapmaları gerekmektedir. Bu sayede daha önceden kaydı olan kullanıcıların profil bilgilerine ulaşılabilmektedir. Sistemde kaydı olan

kullanıcılar bir sorgu ekranı üzerinden, o gün için dinlemek istedikleri müziklerle ilgili girecekleri sorgu kriterlerini kullanabilecekleri gibi, hiçbir sorgu kriteri girmeden de, sistemden öneri yapmasını isteyebileceklerdir.

Tavsiye sistemi kullanıcının sorgusunu girmesi ile başlamaktadır. Sistemde var olan veya sisteme yeni kayıt olmuş kullanıcılar tavsiye motoruna bir öneri yapılması için istekte bulunurlar ve sistem bu kullanıcılara önerilerle dönmektedir. Kullanıcı isteğini yaptığında, kullanıcının sisteme giriş yapmış olduğu kişisel bilgileri (yaş, ülke ve cinsiyet vb), kullanıcı profil veri tabanı üzerinden alınmaktadır. Buna ek olarak, kullanıcın bir öneri için tercih edebileceği bilgiler (örneğin; şarkıcı, şarkı uzunluğu, şarkının çıkış zamanı, şarkının ait olduğu kategoriler ve şarkının dinlenme sayısı) kullanıcıdan isteğe bağlı olarak sorgu ölçütleri içinde alınmaktadır.

Sistem iki farklı tavsiye motorunu (CBR ve CF) desteklemektedir. Bu tavsiye motorlarının üretmiş olduğu sonuçlar, ağırlandırılarak tek bir liste şeklinde son kullanıcıya gösterilmektedir. Ağırlıklandırmanın nasıl yapıldığının detayları aşağıdaki bölümlerde verilecektir. Tavsiye sistemi üzerindeki yöntem Şekil 4.1'de gösterildiği gibi çalışacaktır.

İşbirlikçi filtreleme tavsiye motoru kullanılırken, kullanıcı tabanlı işbirlikçi filtreleme yapılmaktadır. Bundan dolayı sisteme kullanıcının tekil belirleyicisi gönderilmektedir. Daha sonra bu kısımda seçilen işbirlikçi filtreleme benzerlik algoritması benzer kullanıcılardan öneri seçecektir ve sistem kullanıcısına bu önerileri iletecektir. Daha önce vurgulandığı üzere, bu tez kapsamında geliştirdiğimiz tavsiye motoru müzik verisi üzerinde önerilerde bulunmaktadır. Bu yüzden, bu sistemde bahsettiğimiz önerilen ürünler, kullanıcılar tarafından dinlenilen şarkılardır. Geliştirdiğimiz yöntemin desteklediği işbirlikçi filtreleme benzerlik algoritmaları şu şekildedir: Pearson Correlation Benzerlik Algoritması, Euclidean Distance Benzerlik Algoritması, Uncentered Cosine Benzerlik Algoritması, Tanimoto Coefficient Benzerlik Algoritması, City Block Benzerlik Algoritması. Sistemin çalışması sırasında kullanılacak olan benzerlik algoritmalarının hangisi olacağı ise yapılandırma dosyasında belirtilebilmektedir. x ve y iki ayrı kullanıcı olmak üzere 1 ile n arası ürün olan bir sistemde bu iki kullanıcının birbirine olan benzerliklerini bulan algoritmalar aşağıda belirtilmiştir.

Öklid ve Manhattan benzerlik algoritmaları iki nokta arasındaki uzaklığı doğrusal olarak bulan algoritmalardır. Tavsiye sistemlerinde bu iki noktaya, farklı iki kullanıcı karşılık gelmektedir. Kullanılan algoritmaların amacı bu iki kullanıcı arasındaki benzerliği bulmaktır. r bir doğal sayı olmak üzere $r = 1$ için (4.1) eşitliği Manhattan Benzerlik algoritmasını, $r = 2$ için ise Öklid Benzerlik algoritmasını göstermektedir.

$$d(x,y) = (\sum_{k=1}^{n} |x_k - y_k|^r)^{\frac{1}{r}} \quad (4.1)$$

Sistem üzerinden kullanılabilen diğer bir algoritma ise Pearson Korelasyon Benzerlik algoritmasıdır. Bu algoritma iki kullanıcı arasındaki benzerliği (4.2) eşitliğinde gösterilen formülle bulmaktadır.

$$d(x,y) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \quad (4.2)$$

İşbirlikçi filtreleme motorunda kullanılabilen algoritmalardan bir diğeri ise Kosinüs Benzerlik algoritmasıdır. Bu algoritmada verilen iki kullanıcı arasındaki benzerlik (4.3) eşitliğinde gösterildiği gibi bulunmaktadır.

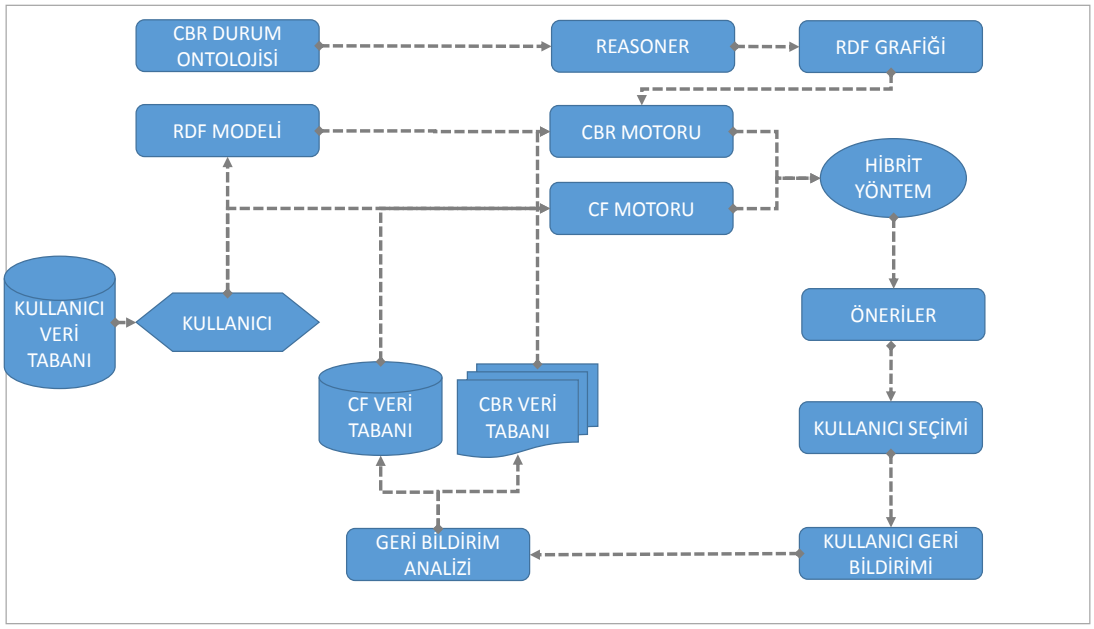
$$d(x,y) = \frac{x.y}{||x||\,x\,||y||} \quad (4.3)$$

Geliştirilen uygulamada kullanılabilen algoritmaların sonuncusu Tanimoto Katsayı Benzerlik algoritmasıdır. Bu algoritma iki kullanıcı arasındaki benzerliği (4.4) eşitliğinde gösterildiği gibi bulmaktadır.

$$d(x,y) = \frac{x.y}{(||x||^2)+(||y||^2)-x.y} \quad (4.4)$$

Kullanıcıdan alınan, profil bilgileri (demografik bilgiler, müzik tercihleri v.b.) geliştirdiğimiz ontolojiye dayalı veri modellemesi kullanılarak RDF üçlülerine dönüştürülmekte ve daha sonra durum tabanlı çıkarsama motoruna sorgu olarak verilmektedir. Durum tabanlı çıkarsama (CBR) motoru, öncelikle elde edilen RDF üçlüleri üzerinde, çıkarsama yaparak, kullanıcı verisinde yer alan ilişkisel verilerin zenginleştirilmesini sağlamaktadır. Çıkarsama yeteneği ile gizli ilişkiler RDF üçlüleri olarak kullanıcıdan elde edilen RDF üçlülerine eklenmektedir. Çıkarsama yönteminin

nasıl uygulanacağı aşağıda detaylı olarak verilmektedir. Elde edilen tüm RDF üçlüleri bir durum nesnesine dönüştürüldükten sonra, CBR motoru, kullanıcı bilgilerinden oluşan durum nesnesiyle, durum verilerinin tutulduğu CBR veri tabanı üzerindeki durum nesneleri arasında karşılaştırma yaparak, benzerliklerin tespitini sağlamaktadır. Karşılaştırmadan elde edilen sonuçlar benzerlik değerlerine göre sıralanmaktadır ve tek bir öneri listesine dönüştürülmek üzere sistemin hibrit modülüne gönderilmektedir. CF yönteminde olduğu gibi CBR yönteminde de kullanılan benzerlik algoritmaları, tavsiye sisteminin yöneticisi tarafından yapılandırma dosyasında seçilebilmektedir.



Şekil 4.1 Oluşturulan hibrit tavsiye sistemi

İşbirlikçi filtreleme ve durum tabanlı çıkarsama tekniklerinden elde edilen sonuçlar, ayrı ayrı kendi içlerinde belli bir benzerlik eşik değerinden yüksek oldukları durumda, aşağıda açıkladığımız ağırlıklandırma yaklaşımına dayalı olarak değerlendirilir ve tek bir liste içinde birleştirilerek kullanıcıya gösterilir. CF ve CBR için kullanılacak olan benzerlik eşik değerleri, yine yapılandırma dosyasında belirlenebilecektir. Kullanıcıdan gösterilen müzik parçalarına göre bir veya birden daha fazla seçim yapması beklenmektedir. Daha sonra kullanıcının yaptığı seçimlere göre tavsiye sisteminin "GERİ BİLDİRİM ANALİZ" bileşeni çalışmaktadır. Geri bildirim analizi bileşeni kullanıcın yaptığı seçimlere ait önerileri toplar ve seçimler CF yönteminin önerilerinden geliyorsa CF veri tabanına,

durum tabanlı çıkarsama motorunun yaptığı önerilerden geliyorsa CBR veri tabanına kaydedilir.

4.4 Önermekte Olunulan Tavsiye Sisteminin Modüler Bileşenleri

Önerdiğimiz tavsiye sistemi Şekil 4.1'de resmedilmektedir. Bu sistemin bileşenleri aşağıda detaylı olarak açıklanmıştır.

4.4.1 CBR Durum Ontolojisi

Bu bileşen kurulmuş olunan sistemdeki tüm problem ve çözümleri içeren bileşendir. Müzik ontolojisi modeli, modelin sınıfları, modelin özellikleri bu bileşen içinde yer almaktadır.

4.4.2 Reasoner

Bu bileşen durum ontolojisi modelini girdi olarak alarak sistem üzerinde inceleme yapar ve tüm sınıfların birbiriyle ilişkilerini çıkarır. Çıkarılan bu ilişkiler durum tabanlı çıkarsamanın var olan örneklemlerden daha farklı örneklemler üzerinde çalışmasını sağlar.

4.4.3 RDF Grafiği

Reasoner bileşeni çıkarsama işlemini yaptıktan sonra durum tabanlı çıkarsama motorunun girdi olarak kullanacağı ontoloji modelini bu bileşenden almaktadır.

4.4.4 RDF Modülü

Bu bileşen öneri yapılacak ürünle ilgili sorgu verilerinin, ontolojiye dayalı olarak RDF üçlülerine dönüştürüldüğü bileşendir. Kullanıcının demografik bilgileri ve tercihleri veri girdisi olarak kullanılmakta, geliştirdiğimiz ontolojiye dayalı olarak, ontolojiye dayalı oluşturulan RDF üçlüleri, bu veri girdileri kullanılarak oluşturulmaktadır. RDF Modülü, kullanıcının sorgusunu aynı zamanda bir CF Motorunun kullandığı sorgu nesne yapısına dönüştürmektedir.

4.4.5 CBR Motoru

Bu bileşen durum tabanlı çıkarsama yöntemimizi gerçeklediğimiz bileşendir. RDF Modülü bileşeninden elde edilen sorgu durum nesneleri, bu motora girdi olarak verilmektedir. Çıktı olarak ise geçmiş deneyimlerden elde edilerek bulunan öneriler alınmaktadır.

4.4.6 CF Motoru

Bu bileşen işbirlikçi filtreleme metotları kullanılarak tavsiyelerin oluşturulduğu modüle verilen isimdir. RDF Modülü bileşeninden elde edilen sorgu nesnesi bu motora girdi olarak verilmektedir. Çıktı olarak ise işbirlikçi filtrelemeden elde edilen öneriler alınmaktadır.

4.4.7 Hibrit Yöntem Bileşeni

Hibrit yöntem bileşeni işbirlikçi filtreleme motoru ve durum tabanlı çıkarsama motorundan elde edilen önerileri, tavsiye sistemi yöneticisi tarafından yapılandırılabilen ağırlıklandırma mekanizmasına göre, sıralar ve elde edilen tavsiyeleri son kullanıcıya bir liste halinde gönderir. Ağırlıklandırma mekanizması aşağıda detaylı olarak açıklanmaktadır.

4.4.8 Kullanıcı Geri Bildirim Modülü

Bu bileşen kullanıcıya önerilen ürünlerden kullanıcının seçtiği ürünlerin alınması ve geri bildirim analiz modülüne gönderilmesine sağlamaktadır. Önerilen tavsiye sistemi için bu bileşen bir web ara yüzü şeklinde tanımlanmıştır.

4.4.9 Geri Bildirim Analiz Bileşeni

Bu bileşen kullanıcıdan alınan bilgilerin analizinin yapıldığı ve işbirlikçi filtreleme ve durum tabanlı çıkarsama veri tabanlarının güncellenmesini sağlamaktadır.

4.4.10 CF Veri Tabanı

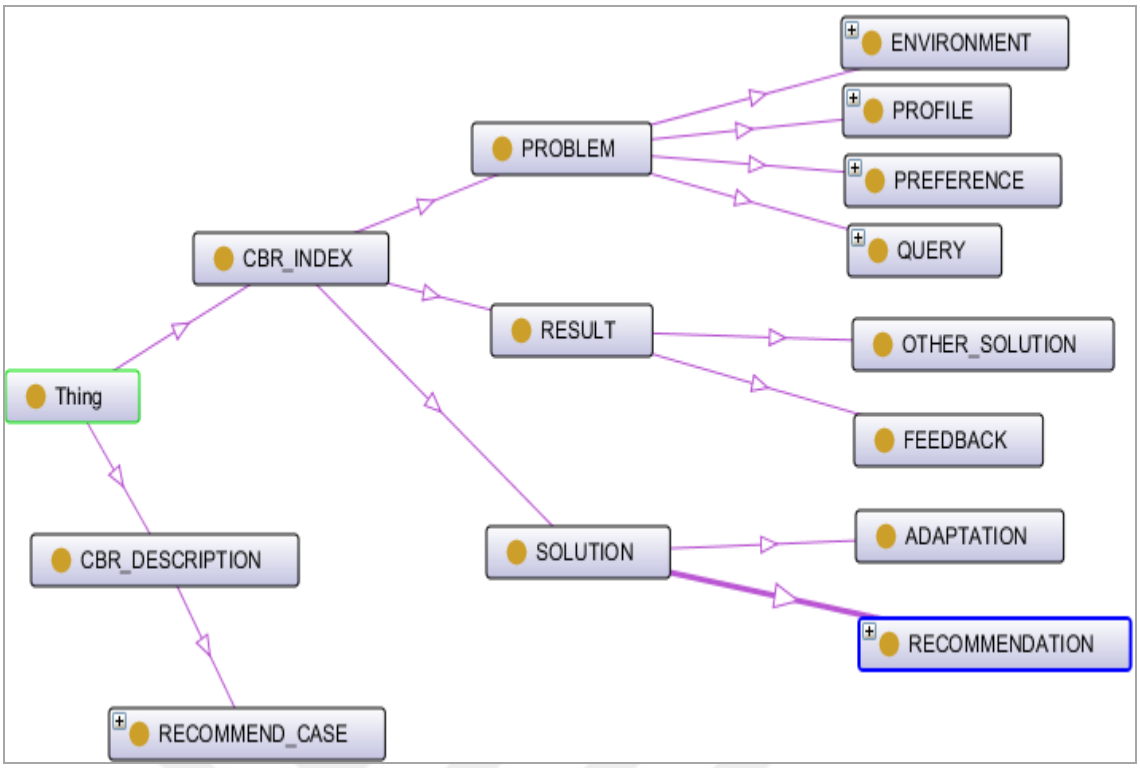
İşbirlikçi filtreleme için işlenmiş ve üzerinden öneri yapılacak verilerin toplandığı veri tabanıdır.

4.4.11 CBR Veri Tabanı

Durum tabanlı çıkarsama için işlenmiş ve üzerinden öneri yapılacak verilerin toplandığı veri tabanıdır. Bu veri tabanında yer alan birim veri, durum nesneleridir. Durum nesneleri ise RDF üçlülerinden oluşmaktadır.

4.5 CBR Veri Tanımlama Modeli

Önerdiğimiz tavsiye sisteminde CBR veri tabanında, verinin tanımlanması için daha önceki bölümde açıkladığımız üzere, bir ontolojiye dayalı olacak gerçekleştirilmektedir. Burada ontolojiye dayalı bir veri modeli seçmemizin ana nedeni, veriden yola çıkarak çıkarımlar yapmak ve daha sağlıklı öneriler sunabilmektir. Bu tez kapsamında müzik verileri üzerinde çalışıldığı için, CBR durumları için oluşturduğumuz ontoloji, kullanıcılarının müzik beğenilerini tanımlayabilecek bilgi varlıklarını içermektedir. Şekil 4.2' de ana ağaç yapısını verdiğimiz ontoloji, CBR durumlarında yer alan RDF üçlülerin oluşturulması için kullanılabilecek bilgi varlıkları ve aralarındaki ilişkileri göstermektedir. Durum tabanlı çıkarsama yönteminde kullanılan her bir duruma ait, problem ve çözüm için ontoloji verileri olan RDF üçlüleri kullanılmaktadır.



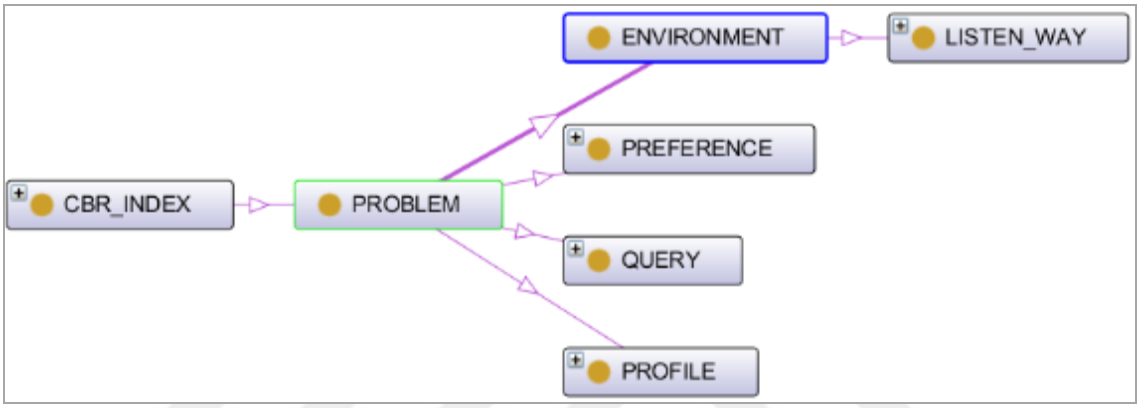
Şekil 4.2 Durum veritabanı ve ontolojiye dayalı oluşturulmuş RDF üçlüleri

Geliştirdiğimiz ontoloji iki ana bölüme sahiptir. “CBR_INDEX”, durum tabanlı çıkarsamadaki durumları modelini gösteren bilgi varlığıdır. “CBR_INDEX” sınıfı üç tane alt sınıf içermektedir. Bunlar “PROBLEM”, “RESULT” ve “SOLUTION” sınıflarıdır. “PROBLEM” sınıfı sorgu parametrelerini içerir ve dört alt sınıftan oluşmaktadır. Bu sınıflar “ENVIRONMENT”, “PROFILE”, “PREFERENCE” ve “QUERY” sınıflarıdır. “RESULT” bölümü ise iki, alt sınıf içerir. Bunlar “OTHER_SOLUTION” ve “FEEDBACK” alt sınıflarıdır. Son bileşen ise “SOLUTION” sınıfıdır. Bu sınıf ise “ADAPTATION” ve “RECOMMENDATION” alt sınıflarını içermektedir.

Önerdiğimiz sistemde CBR veri tabanında yer alan her bir durum, öneri sistemi üzerinden gidilerek bulunmuş ve dinlenmiş bir şarkı ile ilgili bilgileri içermektedir. CBR durumu içinde yer alan “PROBLEM” sınıfı, şarkının öneri sisteminde aranması sürecinde girilmiş sorgularla ilgili bilgileri, kullanıcının profil bilgilerini, önceliklerini/istek bilgilerini, şarkının bulunduğu ortamla ilgili bilgileri içermektedir. “SOLUTION” sınıfı ise, şarkının dinlenme durumu söz konusu ise, bu şarkıyla ilgili bilgileri içermektedir. Bundan dolayı, CBR veri tabanı her bir şarkı dinleme için bir CBR durumu içerecektir. CBR durumu (şarkı dinleme olayı) tavsiye sistemi kullanılarak oluşmuşsa, CBR durumu

içinde yer alan, sorgu (QUERY) sınıfı ve bu sınıfın alt bileşenleri de doldurulmuş olacaktır. Eğer tavsiye sistemi kullanılmadan oluşan dinleme durumları varsa, CBR durumu içinde yer alan öncelik (PREFERENCE) ve demografik bilgiler (PROFILE), bu şarkıyı dinleyen kullanıcının sistemde olan bilgilerinden alınacaktır.

"ENVIRONMENT" sınıfı, "PROBLEM" sınıfın ilk alt sınıfıdır. "ENVIRONMENT" sınıfı müziğin hangi yolla dinlendiğini tutmaktadır. Örnek olarak cd, internet, radio vb. bilgilerini tutmaktadır. Şekil 4.3 "PROBLEM" sınıfının ağaç yapısı gösterimini resmetmektedir.



Şekil 4.3 Bir CBR durumunun Environment sınıfının alt sınıflarının gösterimi

"PROFILE" sınıfı ise, "PROBLEM" sınıfının diğer bir alt sınıfıdır. Bu sınıf altı alt sınıf içermektedir.Şekil 4.4, "PROFILE" sınıfının ağaç yapısını göstermektedir. Bu sınıfın alt sınıflarını aşağıda detaylı olarak açıklıyoruz.

CULTURAL_IDENTIFIER: İnsanların kültürel geçmişleri onların müzik seçimlerini, konuşmalarını ve giyimlerini etkilemektedir [30]. Bundan dolayı CULTURAL_IDENTIFIER sınıfı PROBLEM sınıfının bir alt sınıfı olarak CBR ontolojimize eklenmiştir.

AGE: Yaş ve sosyal statü insanları müzik seçimlerinde önemli bir rol oynamaktadır. Yaşlı ve zengin insanların müzik seçimleri genç ve fakir insanlardan farklıdır [31]. AGE sınıfı bundan dolayı CBR ontolojimizde PROBLEM sınıfının alt sınıfı olarak yer almaktadır.

COUNTRY: Yaşanılan çevre, daha önce yaşanmış şehirler, doğulan şehir ve yaşanılan ülke insanların müzik seçimlerine etki eder [32]. COUNTRY sınıfı bundan dolayı CBR ontolojimizde PROBLEM sınıfının alt sınıfı olarak yer almaktadır.

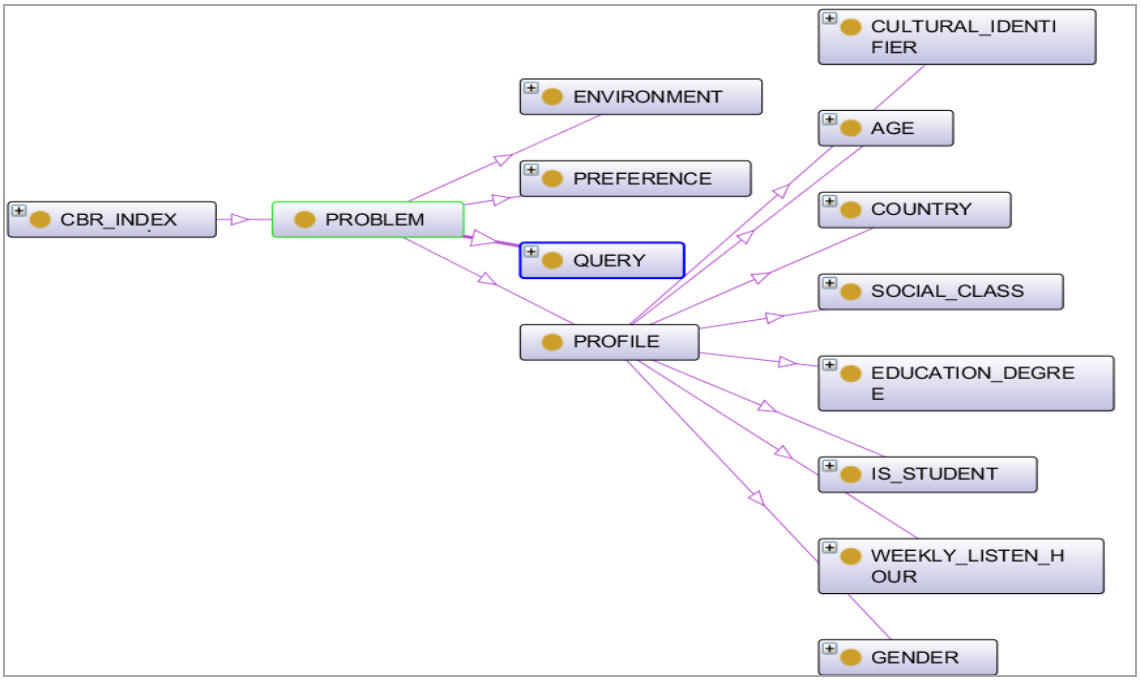
SOCIAL_CLASS: İnsanların ait olduğu sosyal sınıfların (zengin, fakir vs.) bu insanların dinledikleri müziklere etki ettiği gözlemlenmiştir [33]. SOCIAL_CLASS sınıfı bundan dolayı CBR ontolojimizde PROBLEM sınıfının alt sınıfı olarak yer almaktadır.

EDUCATION_DEGREE: İnsanların eğitim derecelerinin dinledikleri müziklere etkisi vardır. Bundan dolayı EDUCATION_DEGREE alt sınıfı PROBLEM sınıfının bir alt sınıfı olarak CBR ontolojimize eklenmiştir.

IS_STUDENT: Yapılan bir çalışma üniversite veya kolejde öğrenci olanların müzik zevkinin diğer insanlara göre daha farklı olduğunu göstermiştir [33]. IS_STUDENT alt sınıfı, bu yüzden, PROBLEM sınıfının bir alt sınıfı olarak CBR ontolojimize eklenmiştir.

WEEKLY_LISTEN_HOUR: İnsanların müzik dinleme süreleri bu kişilerin müzik seçimlerine etki etmektedir. WEEKLY_LISTEN_HOUR alt sınıfı, bu yüzden, PROBLEM sınıfının bir alt sınıfı olarak CBR ontolojimize eklenmiştir.

GENDER: Cinsiyet müzik zevkine etki eden etmenlerden bir tanesidir. Örneğin yapılan bir çalışmada kadınların dans, elektronik, hymns, gospel kategorilerine ait müzikleri seçtiği erkeklerin ise blues, rock, heavy metal ve jazz tercih ettikleri görülmüştür [34]. GENDER alt sınıfı, bu yüzden, PROBLEM sınıfının bir alt sınıfı olarak CBR ontolojimize eklenmiştir.



Şekil 4.4: CBR durumunun profil sınıfının ve alt sınıflarının gösterimi

“PREFERENCE”, “PROBLEM” bileşenin diğer bir alt sınıfıdır. Bu sınıf müzik tercihleri hakkında bilgileri içeren bilgileri tutmaktadır. 8 tane alt sınıfa sahiptir. Şekil 4.5 bu sınıf ve alt sınıflarına ait ontoloji diyagramını göstermektedir.

RECORDING_TYPE (KAYIT TİPİ) : Şarkıların kayıt tipi kullanıcıları müzik seçimlerine etki etmektedir. Plak, kaset, MP3 gibi kayıt formatları kullanıcıların müzik tercihlerinde etkili olabilmektedir. Bu sınıf şarkı kayıt tipi bilgisini tutmaktadır.

THEME (TEMA) : Bir müzik parçasının arka planında kullandığı çalgılar ve yaptığı tonlamaların kullanıcıların müzik seçilerinde etkili olduğu görülmüştür. Bu sınıf müzik parçasının arka planında kullandığı çalgılar ve yaptığı tonlamalar ile ilgili bilgileri tutmaktadır.

ARTIST (ŞARKICI) : Bir şarkıyı söyleyen kişi onu dinleyen kullanıcıların şarkı tercihlerinde çok etkili olmaktadır. Bu sınıf, kişinin şarkıcı tercihlerini tutmaktadır.

COMPLEXITY (KARMAŞIKLIK) : Bir şarkının algılanan karmaşıklığı kullanıcıları tercihlerinde etkilidir. Örneğin yoğun çalgı, sürekli yüksek ses veya sözlerin çok olması gibi etmenler müzik seçimini etkilemektedir. Bu sınıf, şarkının karmaşıklık derecesi ile ilgili tercih bilgisini saklamaktadır.

MOOD (MOD) : İnsanların o an bulundukları ruh hali şarkı seçimlerini etkileyen önemli bir faktördür. Kullanıcının iyi veya kötü bir gününde olması şarkı tercihlerine direkt olarak etki etmektedir. Bu sınıf, kullanıcının bulunduğu ruh haliyle ilgili bilgiyi saklamaktadır.

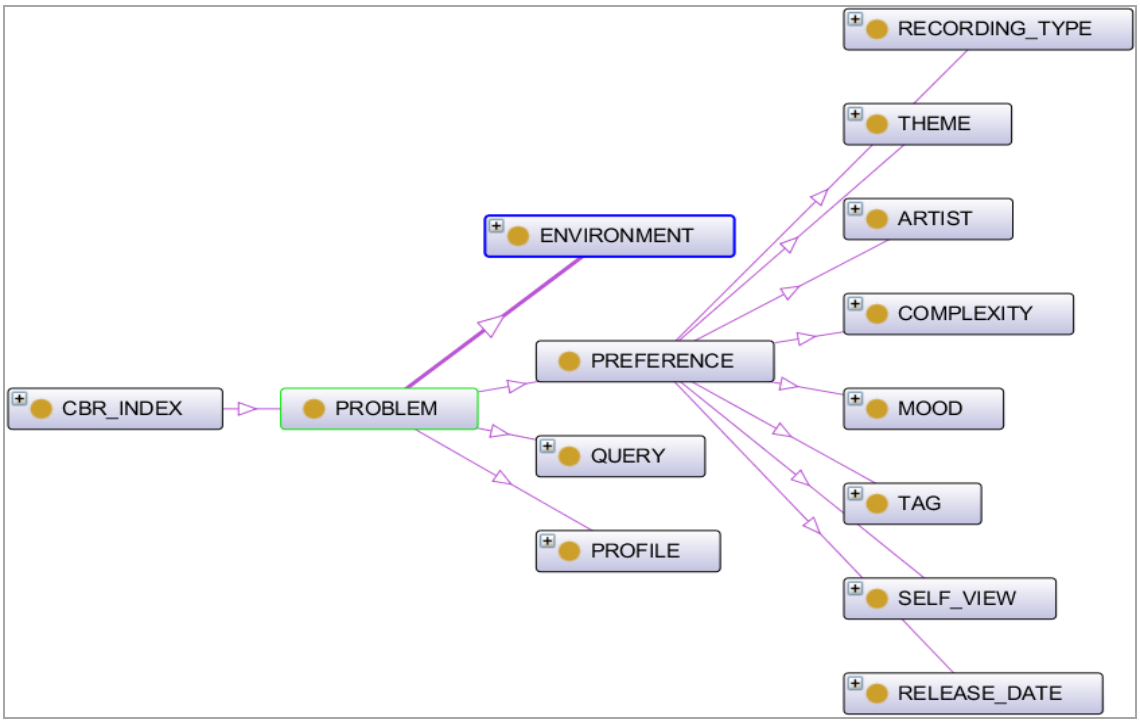
TAG (KATEGORİ) : Şarkıların sahip olduğu müzik kategorileri (rock, jazz, vs.) kullanıcıların müzik tercihlerinde önemli bir etkendir. Bu sınıf müzik kategorisi bilgisini saklamaktadır.

SELF_VIEW (DİNLENME SAYISI) : Bir şarkının diğer kullanıcılar tarafından dinlenme sayısı şarkı seçimi üzerinde etkilidir. Genellikle insanlar daha fazla dinlenen şarkıları tercih etmektedir.

RELEASE_DATE (ÇIKIŞ TARİHİ) : Bir müzik parçasının yayınlanma tarihi bazı kişiler için bir seçim nedeni olabilmektedir. Şarkının ait olduğu dönemde yaşanılan olaylar bu seçimleri tetikleyebilmektedirler.

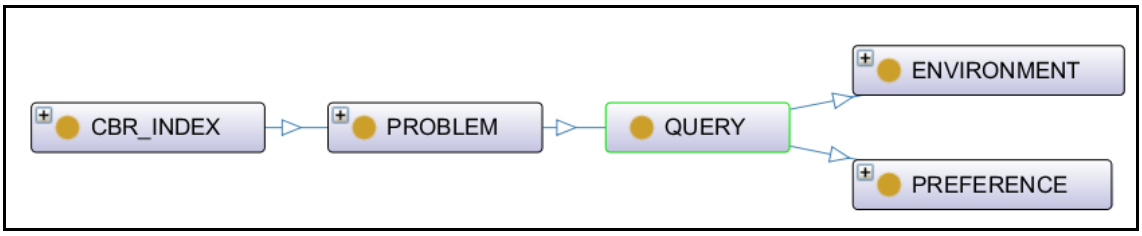
FAMILIARITY (YAKINLIK) : Bir şarkının kullanıcıya hissettirdiği yakınlık bir seçim nedeni olabilmektedir. Şarkıda bulunan bir melodi veya sözlerindeki kültürel yakınlık bu seçime etki edebilmektedir.

SEASON (SEZON): Bir şarkının ait olduğu sezonun kullanıcıların müzik tercihlerinde etkili olduğu görülmektedir. Örneğin bazı insanların yaz mevsimi için çıkan şarkıları daha fazla tercih ettiği görülmüştür.



Şekil 4.5 CBR durumun Preference sınıfının ve alt bileşenlerinin gösterimi

"PROBLEM" sınıfının diğer bir alt sınıfı ise "QUERY" sınıfıdır. Daha önce bahsedilen "PREFERENCE" ve "ENVIRONMENT" sınıfları girilmesi zorunlu olmayan bilgileri içerebilmektedir. Şekil 4.6 "QUERY" sınıfının ontolojik gösterimini içermektedir. Ancak "QUERY" sınıfı, kullanıcı tarafından girilen sorguyu tanımladığı için bilgi girilmesi zorunlu alanları içermektedir. "QUERY" sınıfının alt sınıfları, "PREFERENCE" ve "ENVIRONMENT" sınıflarının alt sınıflarının tümünü içermektedir. "QUERY" oluşturulurken, kullanıcı öneri almak istediği bir şarkı ile ilgili öncelikleri belirleyebilmelidir. Burada sorgu belirlenirken, şarkıyla ilgili tüm beğeni kriteri oluşturabilecek kriterler girdi olarak kullanılabilmektedir.

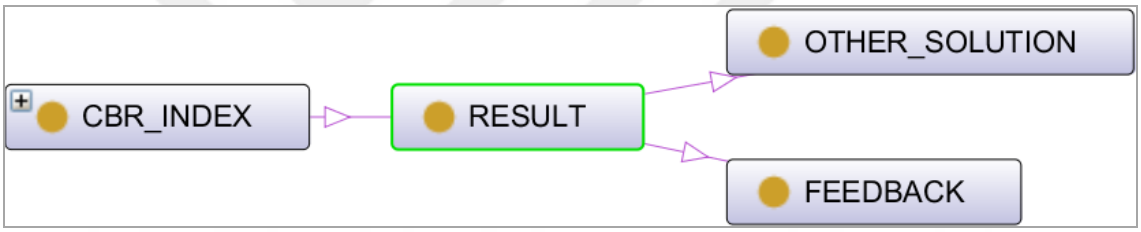


Şekil 4.6 Query sınıfın ontolojik gösterimi

"RESULT" sınıfı iki alt sınıfa sahiptir. Bunlar "OTHER_SOLUTION" ve "FEEDBACK" sınıflarıdır. Şekil 4.7 "RESULT" sınıfının alt sınıflarıyla ilgili detaylar aşağıda verilmektedir.

OTHER_SOLUTION: Kullanıcıya önerilen fakat kullanıcının tercih etmediği önerileri gösteren ontoloji sınıfıdır.

FEEDBACK: Seçilen öneriler hakkında geri bildirim analiz bileşenine bilgi verilmesi amacıyla kullanılan ontoloji sınıfıdır.

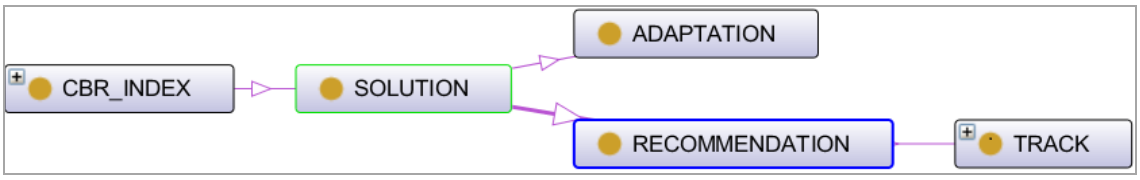


Şekil 4.7 CBR durumunun sonuç ve geri bildirim bileşenlerinin gösterimi

"SOLUTION" sınıfı ise CBR durumundaki çözüm bilgilerini tutan alt sınıftır. İki alt sınıf içermektedir. Bunlar "ADAPTATION" ve "RECOMMENDATION" alt bileşenleridir. "ADAPTATION" alt bileşeni verilen çözümün sisteme adaptasyonunu sağlarken "RECOMMENDATION" alt sınıfı da yapılan çözümü modellemektedir. Şekil 4.8 "SOLUTION" bileşenin ontolojik yapısını göstermektedir.

ADAPTATION: Verilen çözümün sisteme adaptasyonunu sağlayan ontoloji sınıfıdır. Kullanıcıdan alınan geri bildirimlere göre oluşan seçimin veri tabanlarına kayıt modelidir.

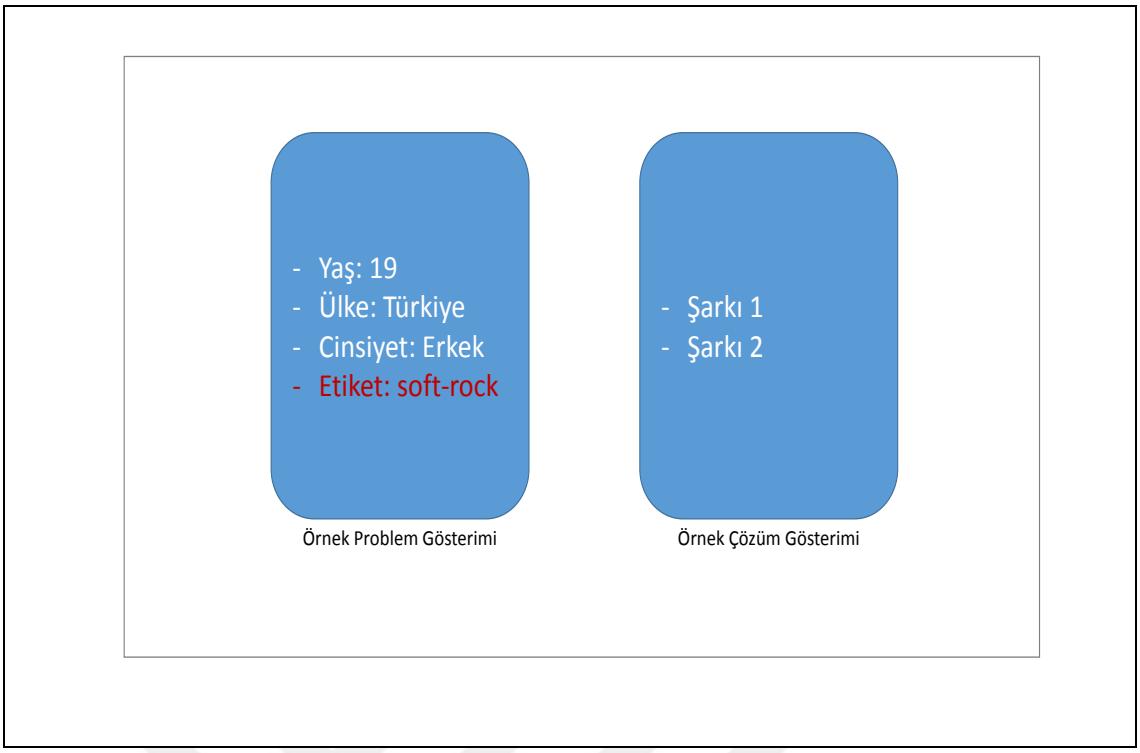
RECOMMENDATION: Kullanıcı tarafından yapılan tercih bir çözüm olarak kabul edilir ve bu ontoloji modeli tarafından temsil edilmektedir.



Şekil 4.8 Çözüm sınıfını gösteren ontoloji modeli

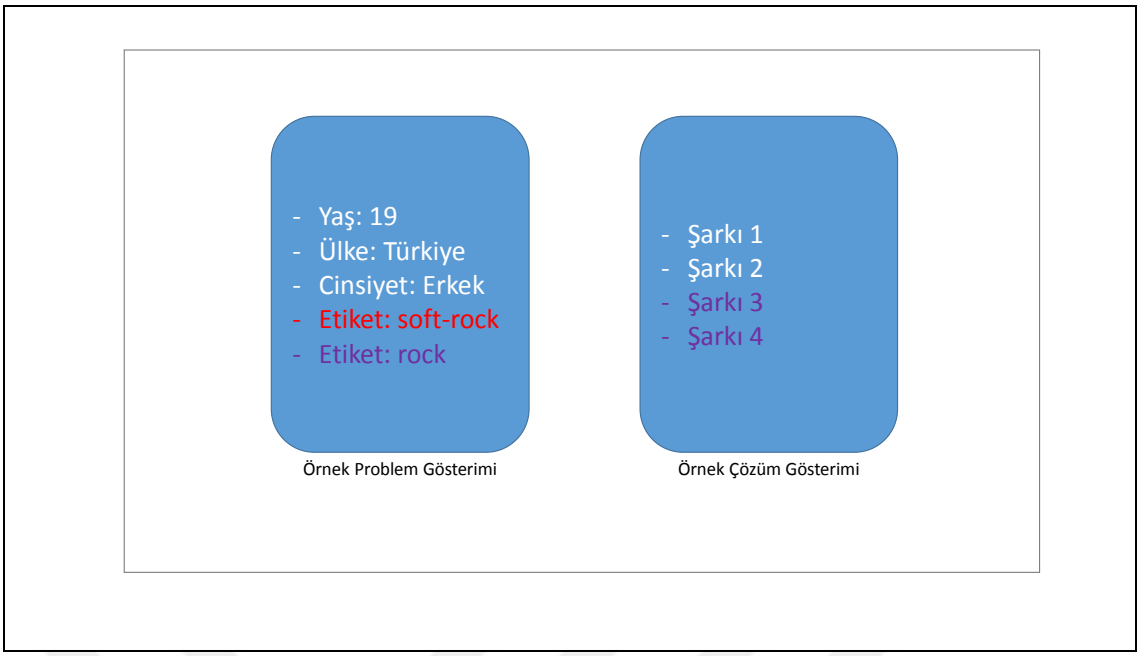
4.6 Durum Tabanlı Çıkarsama (Inference) Yeteneği

Önerdiğimiz yöntem, CBR durumlarını RDF üçlüleri ile tanımlamaya dayanmaktadır. Şekil 2'de verdiğimiz ve önerdiğimiz yöntemin bileşenlerini gösteren resimde yer alan CBR Motoru, kullanıcıdan aldığı sorgu verilerden RDF üçlüleri oluşturmaktadır. CBR Motoru, geliştirdiğimiz CBR Durum Ontolojisi üzerinde çalışan REASONER bileşenini kullanarak, RDF üçlüleri üzerinde çıkarsama yapmaktadır. Bu sayede kullanıcıdan elde edilen sorgu verileri arasında olmayan ancak ontolojiden çıkarsama yoluyla elde edilen gizli bilgiler, sorguyu oluşturan RDF üçlüleri arasına eklenebilmektedir.Şekil 4.9, sorgulama yapılırken, çıkarsama özelliği olmadan sorgulama yapıldığında, CBR motoru tarafından CBR veri tabanından bulunan bir CBR durumunu resmetmektedir. Bu örnekte kullanıcı ilgilendiği müziklerin etiketi olarak "soft-rock", etiketini seçmiştir. Sonuç olarak, kendi demografik bilgilerine benzer nitelikteki kullanıcıların dinlediği "soft-rock" etiketli şarkılar çözüm olarak gösterilmiştir.



Şekil 4.9 Çıkarsama harici CBR motorundan elde edilen önerinin gösterimi

Eğer sonuç, çıkarsama yeteneği kullanılarak elde edilecek olursa, ontolojide yer alan hiyerarşik yapıya dayalı olarak daha fazla gizli ilişkilere ulaşılabilecek ve dolaylı olarak daha fazla çözüm kümesine ulaşılabilecektir. Şekil 4.10'de çıkarsama yeteneği kullanılarak elde edilen çözüm önerisi gösterilmektedir. Buradaki örnekte, kullanıcının yaptığı sorgulama da "soft-rock" müzik kategorisi verildiği halde, çıkarsama yeteneği ile (Ontoloji hiyerarşisinde yer alan ilişki ağından kalıtım özelliğinden yararlanılarak) "soft-rock" kategorisinin üst sınıfı olan "rock" kategorisine ulaşılmış ve sorguya dahil edilmiştir. Bu sorgu ise şekilde de görüleceği üzere çözüm önerilerini arttırmaktadır.



Şekil 4.10 Çıkarsama ile birlikte CBR motorundan elde edilen önerinin gösterimi

4.7 Ağırlıklandırma Yaklaşımı

Önerdiğimiz hibrit yaklaşım yeni kullanıcılara öneri yapamama durumunun önüne geçilmesini ve yapılan önerilerin kalitesinin arttırılmasını sağlamayı amaçlamaktadır. Bu yaklaşımda iki farklı tavsiye motorundan gelen sonuçların birleştirilerek son kullanıcıya sunulması söz konusudur. Bunu yaparken bir ağırlıklandırma yaklaşımı kullanılmaktadır. Ağırlıklandırma yapılırken, sistemlerin önerilerine verilecek önem değerlerinin (ağırlıklar) tamamen bir yapılandırma dosyasında yapılandırılabilir bir şekilde olması tasarlanmıştır. Ağırlıklandırma yapılırken, karşımıza üç farklı durum çıkmaktadır. Birinci durumda, geçmişte sistem tarafından beğenileri çok iyi bilinen kullanıcılara (sistemi daha önce çok kullanmış kullanıcılar), yapılacak önerilerde, daha çok işbirlikçi filtreleme tavsiye motoru kullanılarak öneri verilebilmesine öncelik tanınabilmektedir. İkinci durumda, sistemin hiç tanımadığı kullanıcılara daha büyük ağırlıkla CBR tavsiye motorunun önerilerde bulunması, CBR önerilerine yapılandırma dosyasında daha fazla önem (ağırlık) verilerek tanımlanabilmektedir. Üçüncü durumda ise, sistemde profili olan ve kısmi olarak beğenileri bilinen tanınan ancak, geçmişteki beğenilerinde de eksik veriler olan kullanıcılara, her iki arama motorundan eşit ağırlıklı olarak öneriler yapabilmesine olanak sağlanmaktadır. Tavsiye motorlarından elde edilen önerilerin hibrit ağırlıklandırma sürecine gönderilmeden de bir filtrelemeden geçmesi

yöntemimizin bir parçasıdır. Burada da her iki tavsiye motorunda da ayrı ayrı elde edilmiş önerilerin belli bir benzerlik eşik değerinin üzerinde olması gerekmektedir. Bu eşik değeri de yapılandırma dosyası üzerinden yapılandırılabilir bir değerdir.

4.8 Hibrit Yaklaşım

Önerdiğimiz hibrit öneri sistemi yaklaşımda işbirlikçi filtreleme ve durum tabanlı çıkarsama metotları birlikte kullanılmaktadır. İşbirlikçi filtrelemenin kullanılmasındaki amaç benzer beğenilere sahip kullanıcılar üzerinden giderek, sorguyu yapan kullanıcıya, yapılacak olan önerilerin çeşitliliğini arttırmaktır. Durum tabanlı çıkarsama metotlarının kullanılmasındaki avantaj ise geçmişteki gerçekleşmiş olaylardan yola çıkan bilgi tabanlı bir yöntemin kullanılması ve geçmiş deneyimlerden elde edilebilecek öneriler sunmasıdır. Bu hibrit yaklaşımla, işbirlikçi filtreleme' de karşılaştığımız, sisteme yeni olan kullanıcılara yapılacak önerilerde oluşan soğuk-başlangıç problemi giderilmektedir. Sistem tarafından daha önceki beğeni bilgisi olmayan kullanıcılara, kullanıcının demografik bilgileri ve önceliklerine dayalı olarak, geçmişteki benzer kullanıcılara yapılmış öneri deneyimleri kullanılarak, başarılı öneriler yapılabilmektedir. Önerdiğimiz hibrit yaklaşım her iki önerme yönteminden gelen sonuçları ağırlıklandırarak tek bir liste halinde son kullanıcıya sunmaktadır.

Önerdiğimiz hibrit yaklaşımı, öneri sistemlerinde karşılaşılan veri eksikliği problemine, çıkarsama (inference) yeteneği sayesinde, bir çözüm sağlamaktadır. Buradaki hibrit öneri yaklaşımının RDF üçlüleri üzerinde çalışıyor olması ve herhangi bir alan için oluşturulmuş CBR Durum ontolojisiyle birlikte çalışabilir yapıda olması, sistemin alan-bağımız bir yapıda olmasını sağlamaktadır.

4.9 Hibrit Yaklaşım Algoritması

Önerilen metodolojiyi geliştirmek için gerçekletirilen algoritma Çizelge 4.2' de gösterilmektedir. Geliştirilen algoritmada ilk olarak kullanıcının tekel belirteci, eşik değerleri, kullanıcı-şarkı kayıt kümesi ve yapılacak öneri sayısı girdi olarak alınmıştır. Daha sonra kullanıcın test kümesindeki girilen kullanıcıya ait kullanıcı geçmişi alınmaktadır. Daha sonra ise eğitim kümesinde aynı kullanıcı ait bütün geçmiş veri bulunur. Daha sonra bu iki veriden elde edilen sayılar ile kullanıcı geçmiş oranı

bulunmaktadır. Daha sonra elde edilen veriler ile cf motoru ve cbr motoru çalıştırılarak kullanıcıya önerilen cf sonuçları ve cbr sonuçları bulunur. Bulunan bu sonuçlar başarım değerlerine göre sıralı olarak gelmektedirler.

Algoritmanın başarımı belirleyen en önemli unsur (soğuk-başlangıç problemine üretilen çözüm) test verisinin boyutune göre algoritmanın oranları belirleyerek en doğru sonucu getirmesidir. Kullanıcı geçmişinin az olduğu durumlarda durum tabanlı çıkarsama motoru daha iyi sonuçlar verirken, kullanıcı geçmişinin artmasıyla işbirlikçi filtreleme motoru daha iyi sonuçlar vermeye başlamaktadır. Alınan sonuçlar daha sonra elde edilen oranlara göre birleştirilmektedir ve kullanıcının seçimine sunulmaktadır.

Algoritmada kullanılan diğer bir özellik ise kullanıcı dinlemelerinin normalleştirilerek işbirlikçi filtreleme motorunun kullanabileceği hale getirmektir. İşbirlikçi filtreleme algoritmaları kullanıcı-ürün önerisi yapmak için kullanıcının ürünler için verdiği puanları kullanmaktadır. Elimizdeki ham veride böyle bir veri bulunmamaktadır. Elimizdeki veri bir kullanıcının verilen tarihler arasında bir şarkıyı kaç kere dinlediğidir. Ancak bazı kullanıcılar bir şarkıyı 1 kere dinlerken bazı kullanıcılar aynı şarkıyı 1000 ve üzeri dinlemişlerdir. Bu veriyi puan verisi olarak kullanmak gerektiğinden şarkı dinleme verisini normalleştirmek gerekmiştir. Bu normalleştirmeyi yapmak için logaritma fonksiyonu kullanılmıştır. Kullanıcının bir şarkıyı dinleme sayısı logaritma 2 tabanında normalleştirilmiştir. X dinleme sayısı için normalleştirme (4.5) eşitliğinde gösterilmiştir.

$$f(x) = \begin{cases} 10, & x > 1024 \\ \log_2 x, & x \geq 0 \; ve \; x \leq 1024 \end{cases} \tag{4.5}$$

Algoritmaya eklenen formül sayesinde kullanıcı seçimleri 1 ile 10 arasında normalleştirilmiştir ve işbirlikçi filtreleme motoru bu veri üzerinden çalışmaktadır.

Çizelge 4.2 Hibrit yaklaşım algoritması

Girdi: 1. Öneri yapılacak kullanıcının tekil idsi (kullanıcı-id)

2. Kayıt Kümesi

3. Eşik Değerleri (ALT-ESIK, UST-ESIK) (Sıralama Sonuçlarını Belirlemek İçin)

4. Öneri Sayısı

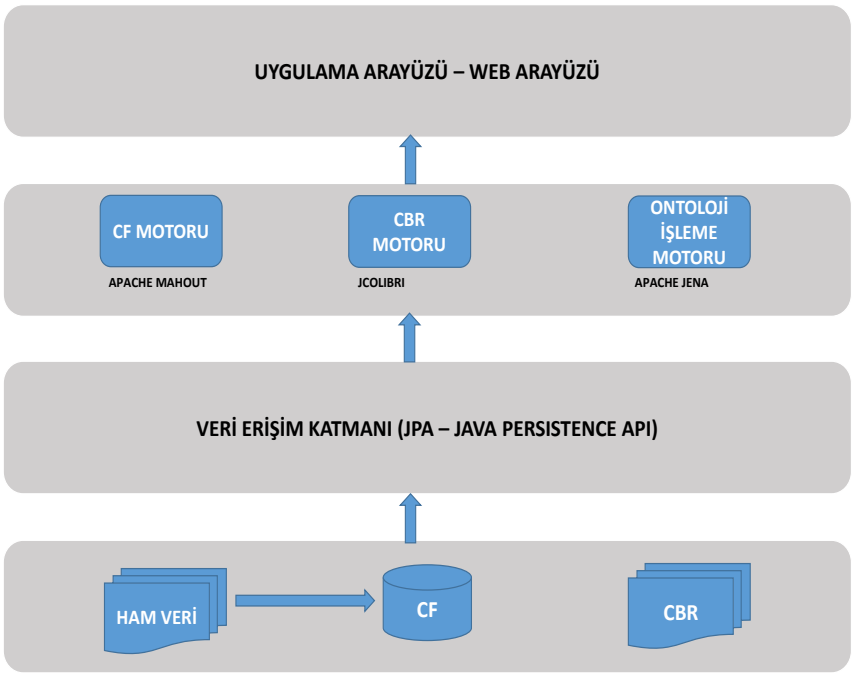
Output: Önerilerin ağırlıklarına göre sıralandırılması

```
KULLANICI-GECMIS-SAYISI = kullanici-sarki-gecmisi-sayisini-bul(kullanıcı-id, test-kümesi),
MAX-KULLANICI-GECMIS-SAYISI = max-kullanici-sarki-gecmisi-sayisini-bul(kullanıcı-id, eğitim-kumesi);
KULLANICI-GECMIS-ORANI = KULLANICI-GECMIS-SAYISI / MAX-KULLANICI-GECMIS-SAYISI;
CF-SONUCLARI = cf-sonuclarini-bul (CF-Veriseti, kullanıcı-id, öneri-sayisi);
CBR-SONUCLARI = cbr-souclarini-bul (CBR-Veriseti, kullanıcı-id, öneri-sayisi);
X = 0; //CF AGIRLIK
Y = 0; //CBR AGIRLIK
If (KULLANICI-GECMIS-ORANI == 0) then {
 //CBR: SON DERECE ONEMLI
 X = 0; Y=100;
} else if (0 < KULLANICI-GECMIS-ORANI < ALT-ESIK) then {
 //CBR: COK ONEMLI
 X = 10; Y=90;
} else if (ALT-ESIK <= KULLANICI-GECMIS-ORANI < UST-ESIK) then {
 //CBR & CF: ESIT ONEMDE
 X = 50; Y = 50;
} else if (UST-ESIK <= KULLANICI-GECMIS-ORANI < 1) {
 //CF: COK ONEMLI
 X = 90; Y = 10;
} else if (KULLANICI-GECMIS-ORANI == 1) then {
  //CF: SON DERECE ONEMLI
  X = 0; Y = 100;
}
HIBRIT-SONUCLARI = BIRLESTIR (EN-YUKSEK X% CF-SONUCLARI, EN-YUKSEK Y% CBR-SONUCLARI);
```

BÖLÜM 5

SİSTEM MİMARİSİ

Sistem mimarisi bölümünde tavsiye motoru uygulamasının yazılım mimarisi üzerinde durulacaktır. Şekil 5.1 öneri sistemi bileşenlerinin yapısını göstermektedir.



Şekil 5.1 Sistem mimarisi

Tavsiye sistemi uygulaması dört ana katmandan oluşmaktadır. Bunlar veri katmanı, veri erişimi katmanı, uygulama katmanı ve kullanıcı ara yüzü katmanıdır.

5.1 Veri Katmanı

Uygulamanın en alt kısmında veri katmanı bulunmaktadır. Veri katmanı uygulamaya işbirlikçi filtreleme verisi, durum tabanlı çıkarsama verisi, sonuç çıkarılmış durum tabanlı çıkarsama verisi ve işlenmemiş veri sağlar.

Durum tabanlı çıkarsama veri tabanı kullanıcıların geçmiş datalarından oluşan problem ve çözümleri içeren bir owl dosyasıdır. Owl dosyaları ontolojileri semantik ilişkilerle birlikte tutan dosyaya verilen addır. Sistemdeki tüm kullanıcıların geçmiş müzik tercihleri bu dosya üzerinde tutulmaktadır. Lastfm müzik sitesinin 1000 kullanıcı için işlenmiş verisi 17 milyon işlemden oluşmaktadır. Bu da yaklaşık olarak 17 milyon problem ve bunların seçilmiş çözümleridir. Bu veriler filtrelendikten sonra durum tabanlı çıkarsama veri tabanına kaydedilmiştir.

Durum tabanlı çıkarsama veri tabanı kullanılarak zenginleştirilmiş yeni bir veri tabanı daha oluşturulmuştur. Lastfm verisinin içindeki şarkıların ait olduğu kategoriler vardır. Bu kategoriler arasındaki ilişki durumlarına göre listelendiğinde ontolojik ilişkilerden yeni çıkarımlar yapılabilmektedir. Örneğin rock kategorisi hard-rock ve soft-rock kategorilerinin üst kategorisi durumundadır. Kullanıcı hard-rock kategorisinde bir arama yaptığında durum tabanlı çıkarsama veri tabanından hard-rock ve bunun üst kategorisi olan rock kategorisi aranacaktır.

Son olarak lastfm verilerinin işlenmemiş şekilde tutulduğu “raw data” veri tabanı vardır. Bu veri tabanı lastfm uygulamasından dosya olarak alınmıştır ve veri tabanına aktarılmıştır.

5.2 Veri Erişimi Katmanı

Veri erişimi katmanı uygulamanın tüm data kaynaklarına ulaşımı yapmasını sağlayan katmandır. Bu katman oluşturulmuş tüm veri tabanlarına erişimi Java Persistence Api (JPA) üzerinden sağlamakla görevlidir. Veri kaynakları normalizasyon yapıldıktan sonra veri tabanlarına kaydedildiği için JPA bunlara erişimi kolaylıkla sağlamaktadır. Tablo isimleri ve kolon adları kullanılarak sınıflar oluşturulur ve bu sınıflar veri tabanı bağlantısı yapıldıktan sonra nesne oluşumlarını otomatik olarak yaparlar.

5.3 Uygulama Katmanı

Uygulama katmanı kısmı öneri sisteminin ana yapısını oluşturmaktadır. Bu kısımda işbirlikçi filtreleme motoru, durum tabanlı çıkarsama metodu, ontolojik işlemleri yapan uygulama ve melez bir yöntemle bunları kullanan ana uygulama vardır.

İşbirlikçi filtreleme motoru öneri hesaplamaları için "Apache Mahout" [35] çatı uygulamasını kullanmaktadır. Kullanıcıları, bunların dinlediği tekil şarkıları ve bunların ne kadar dinlendiğini tutan dosya, bu uygulamaya girdi olarak verildiğinde istenilen algoritmalara göre tavsiyeler üretmektedir.

Ontolojik işlemler için "Apache Jena" [36] çatı uygulaması kullanılmaktadır. Kullanıcıdan alınan demografik bilgi ve müzik tercihlerine göre ontoloji oluşturma işlemi uygulamanın bu katmanında yapılmaktadır. Ayrıca müzik kategorilerine göre yeni ontolojik ilişkilerin keşfi de bu katmanın diğer görevidir. Bu katmanın çıktısı durum tabanlı çıkarsama motoruna verilmektedir.

Durum tabanlı çıkarsama motoru "JColibri" [37] altyapısını kullanmaktadır. JColibri uygulaması verilen girdiye göre geçmiş deneyimlerden, yeni problemlere çözüm üretilen bir araçtır. Ontoloji katmanında üretilen geçmiş deneyimler çıktısı durum tabanlı çıkarsama metodunun girdisi olacaktır ve benzer problemlere göre tavsiyeler üretecektir.

5.4 Kullanıcı Ara Yüzü Katmanı

Kullanıcı ara yüzü katmanı bir web uygulaması şeklindedir. Spring çatısı kullanılarak oluşturulan bu katman kullanıcıların sisteme kayıt olabildiği, sisteme giriş-çıkış yapabildiği ve öneri sistemi sayfalarından oluşmuştur. Bu sayfalar HTML5, CSS3, Javascript gibi modern web teknolojileri kullanılarak tasarlanmıştır. Sisteme ilk defa gelen kullanıcı yaş, ülke ve cinsiyet gibi bilgilerini girdikten sonra sisteme kaydolacaktır. Daha sonra bu bilgilerle sisteme giriş yapıp tavsiye motoru çalıştırılabilir.

Tavsiye motoru sayfasında kullanıcının tercihlerini girebildiği, kategori, şarkı yılı, şarkı süresi vb. filtrelemeleri yapabildiği, yapılan tavsiyeleri görüp, şarkıları dinleyebildiği ekranlar olarak tasarlanmıştır.

BÖLÜM 6

UYGULAMA DETAYLARI

Tavsiye uygulaması yapılırken ilk olarak işlenmemiş veri işlenerek kullanılabilir veri haline getirilmiştir. Lastfm uygulamasının sunduğu işlenmemiş verinin istatistikleri şu şekildedir:

- Toplam satır sayısı: 19.150.868
- Tekil kullanıcı sayısı: 992
- Verilerin bulunduğu tarih aralığı: 2005 – 2009

İşlenmemiş veri dosya şeklinde tutulmaktadır ve yaklaşık 2GB boyutunda bir dosyadan oluşmaktadır. Verinin dosya içinde yapısı ise Çizelge 6.1' de gösterilmektedir.

Çizelge 6.1 Ham müzik verisinin dosya içindeki yapısı

userid \t timestamp \t artistId \t artistName \t trackId \t trackName

Bu veriler tab karakteri ile ayrılmıştır. Bu verinin yanında kullanıcı bilgilerini içeren diğer bir dosya daha vardır. Bu dosyanın yapısı ise Çizelge 6.2' de gösterilmektedir.

Çizelge 6.2 Ham kullanıcı verisinin dosya içindeki yapısı

userid \t gender \t age \t country \t signup

Bu işlenmemiş veriler ilk olarak bir veri tabanı oluşturularak onun içine atılmışlardır. Veri tabanı olarak açık kaynak olmasından ve ilişkisel veri tabanlarına destek verdiği için "MySQL" veri tabanı seçilmiştir. İlk olarak aşağıdaki komut kullanılarak uygulamada

kullanılacak olan veri tabanı oluşturulmuştur. Bu adımdan sonra işlenmemiş veri bu yeni veri tabanına aktarılmıştır. Bu aktarma işlemi aşağıdaki kod ile yapılmaktadır. Aktarma işlemi yapıldıktan sonra tabloya tekil id atama işlemi de yapılmaktadır. Anlatılan veri işlemleri ve sorgular Çizelge 6.3' de göserilmektedir.

Çizelge 6.3 Ham müzik verisinin veri tabanına aktarılması

```
CREATE DATABASE recommendation_engine
USE DATABASE recommendation_engine;
LOAD DATA LOCAL INFILE 'C:/lastfm.tsv' INTO TABLE lastfm;
ALTER TABLE lastfm ADD id INT( 10 ) NOT NULL AUTO_INCREMENT PRIMARY KEY FIRST;
```

Daha sonra aktarılan bu veri normalizasyon işlemine tabi tutulmuştur. İlk olarak veriden tekil parçalar ve tekil şarkıcılar ayrılmıştır. Bunlar için iki ayrı tablo daha oluşturulmuştur. Bunlar "track" ve "artist" tablolarıdır. İşlenmemiş verinin atıldığı "lastfm" tablosundan bu ayrım işlemini yapmak için aşağıdaki kod parçacıkları kullanılmıştır. Track tablosunu oluşturmak için kullanılan sorgular Çizelge 6.4' de gösterilmektedir.

Çizelge 6.4 Track tablosu oluşturma

```
CREATE TABLE track SELECT DISTINCT trackId, trackName FROM lastfm;

ALTER TABLE track ADD id INT(10) NOT NULL AUTO_INCREMENT PRIMARY KEY FIRST;

UPDATE lastfm, track SET lastfm.trackId = track.id where lastfm.trackId = track.trackId;
```

Kullanıcılar için "lastfm_users" adında bir tablo oluşturulmuştur. Bu tablo işlenmemiş veri ile birlikte gelen "users" dosyası kullanılarak hazırlanmıştır. Daha sonra da bu tablo kullanılarak "lastfm" tablosundaki kullanıcı kolonu "users" tablosundaki tekil id ile eşleştirilmiştir. Bu işlemleri gerçekleştirmek için kullanılan kod bloğu Çizelge 6.5' de gösterilmektedir.

Çizelge 6.5 Ham kullanıcı verisinin veri tabanına aktarılması

```
USE DATABASE recommendation_engine;

LOAD DATA LOCAL INFILE 'C:/users.tsv' INTO TABLE lastfm_users;

ALTER TABLE lastfm_users ADD id INT(10) NOT NULL AUTO_INCREMENT PRIMARY KEY FIRST;

UPDATE lastfm, lastfm_users SET lastfm.userid = lastfm_users.id WHERE lastfm.userid = lastfm_users.userid
```

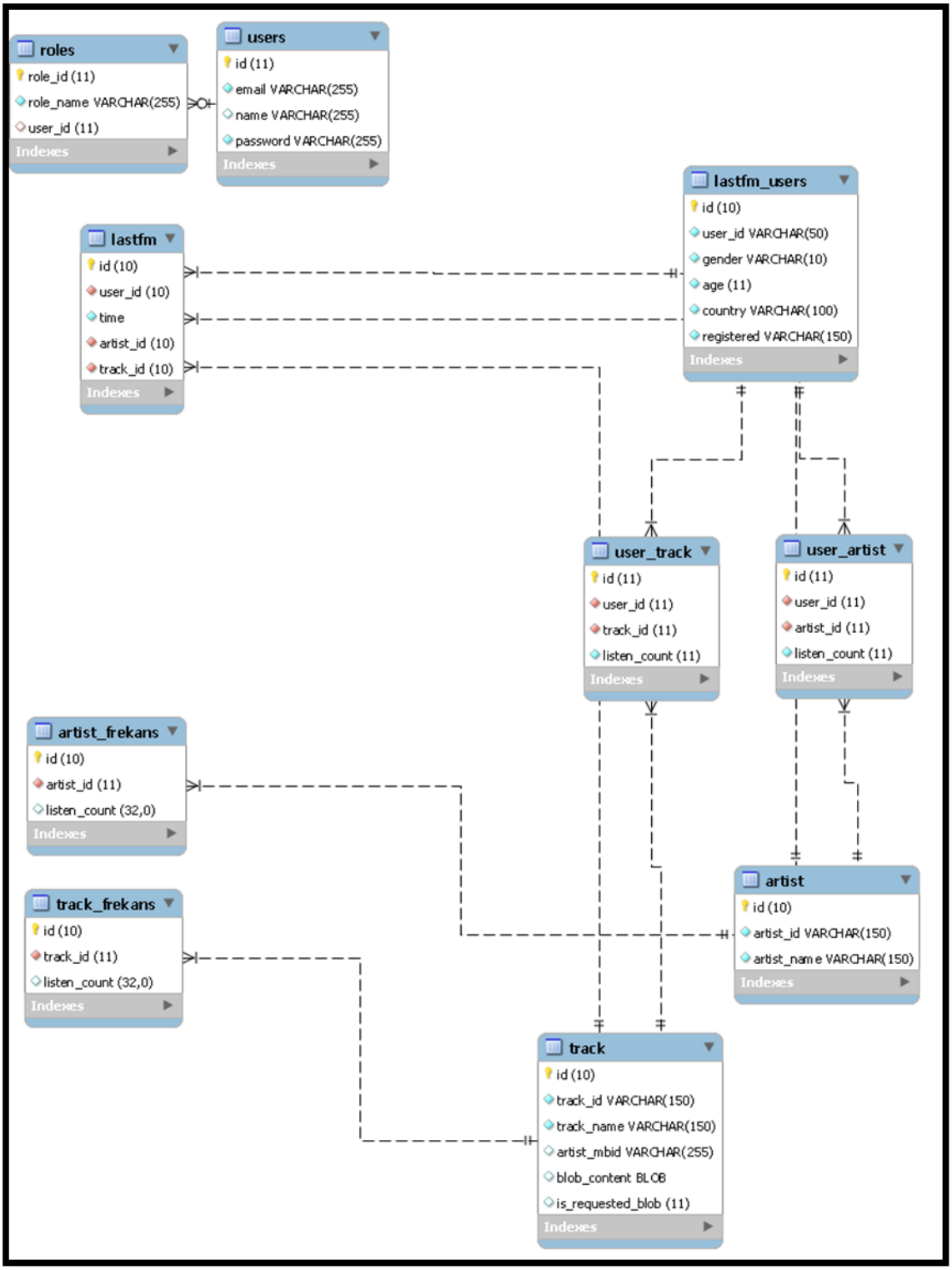
Bu tabloların yanı sıra işbirlikçi filtrelemede kullanmak, veri istatistikleri oluşturmak ve veriye kolay erişim için yeni tablolar oluşturulmuştur. Kullanıcı - şarkı dinleme istatistiklerini tutan “user_track” ve kullanıcı – şarkıcı istatistiklerini tutan “user_artist” tabloları oluşturulmuştur. Ayrıca şarkıya ve artiste bağlı frekans bilgilerini tutmak amacıyla da “track_frekans” ve “artist_frekans” tabloları oluşturulmuştur. Bu tabloların oluşturulmasında kullanılan kod blokları Çizelge 6.6’da anlatıldığı gibidir.

Çizelge 6.6 Kullanıcı – Şarkı tablosunun oluşturulması

```
-CREATE TABLE user_track SELECT userid, track, count(artist) AS listen_count FROM lastfm GROUP BY userid, track;

-ALTER TABLE user_track ADD id INT NOT NULL AUTO_INCREMENT PRIMARY KEY FIRST;
-UPDATE user_track, lastfm_users SET user_track.user_id = lastfm_users.id WHERE user_track.user_id=lastfm_users.user_id;
-UPDATE user_track, artist SET user_track.track =track.id where user_track.track =track.track;
```

Oluşturulan tablolar çok fazla bilgi içerdiğinden dolayı veriye erişim sağlarken yavaşlık problemi oluşmaktaydı. Bu sorunu çözmek için veri tabanına indekslemeler yapılmıştır. Lastfm tablosundaki “track_id”, “artist_id”, “user_id” gibi kolonlara eklenen indekslerle aramalar hızlandırılmış ve erişim problemi optimize edilmiştir. Oluşturulan veri tabanın şeması Şekil 6.1’de gösterilmektedir.



Şekil 6.1: Veri tabanı şeması

Veri temizleme, filtreleme ve normalizasyon işlemleri yapıldıktan sonra daha sonra uygulamanın veri erişimi katmanı yazılmaya başlanmıştır.

Veri erişimi için ise Java Persistence Api (JPA) kullanılmaktadır. Bu teknoloji veri tabanındaki alanları sınıftaki nesne özellikleri eşleştirir ve veri tabanı işlemlerini ve

veriye erişimi kolaylaştırır. Örneğin “user_track” tablosu için tanımlanan sınıf Çizelge 6.7’de aktarılmıştır.

Çizelge 6.7 Örnek bir nesne gösterimi

```
@Entity
@Table(name="user_track")
public class UserTrack {
   @Id @GeneratedValue(strategy=GenerationType.AUTO)
   private int id;
   @ManyToOne
   @JoinColumn(name = "user_id")
   private DataUser user;

   @ManyToOne
   @JoinColumn(name = "track_id")
   private Track track;

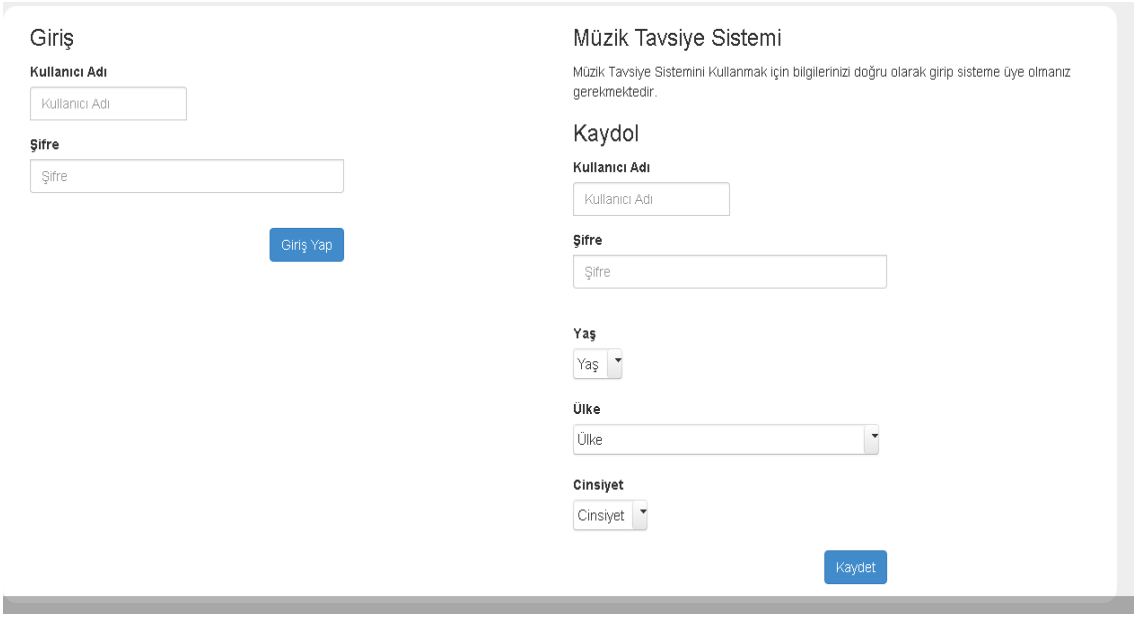
   @Column(name="listen_count")
   private int listenCount;

   public int getId() {
      return id;
   }
   public void setId(int id) {
      this.id = id;
   }
   public int getListenCount() {
      return listenCount;
   }
   public void setListenCount(int listenCount) {
      this.listenCount = listenCount;
   }
}
```

Tavsiye uygulamasında veri erişimi çok sık kullanıldığından üstteki gibi bir yapıya ihtiyaç duyulmuştur. Bu yeni teknoloji ile birlikte veri tabanı alanlarını eşleme her kolonun adını tek tek yazar değil sadece tek bir sınıfta tanımlanarak uygulamada sağlamlık, tekrar kullanılabilirlik ve veri erişimi kolaylığı sağlanmıştır.

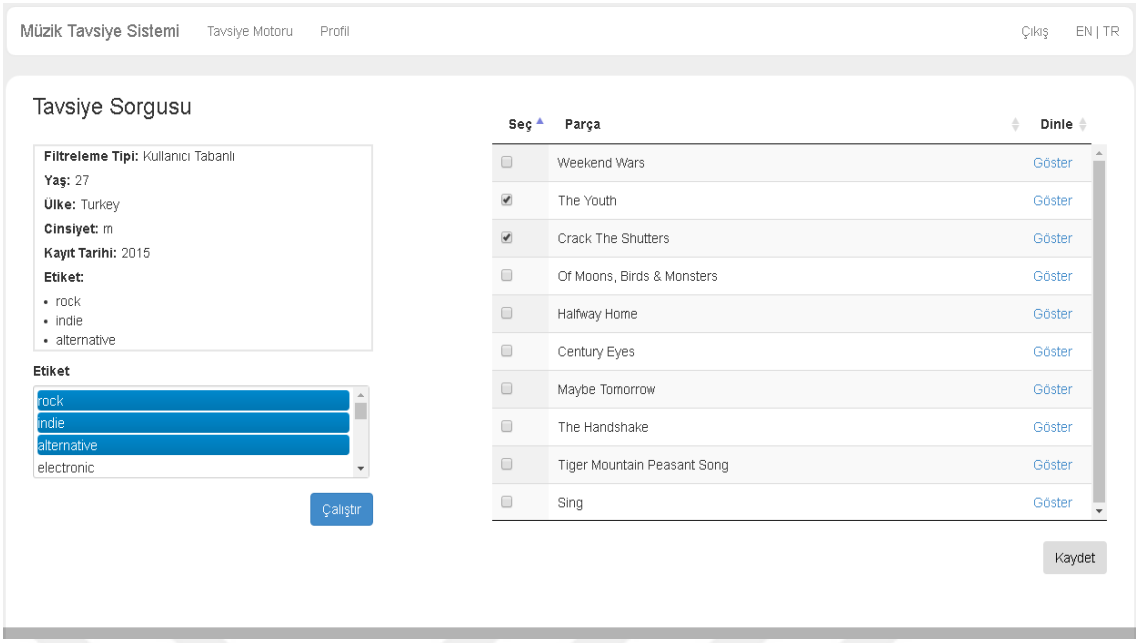
Bu kısımdan sonra uygulama kullanıcı ara yüzü katmanı birlikte kodlanmıştır. Bu kısımlar birbirleriyle paralel ilerlemeleri gerektiği için ve yapılan geliştirmeler birbirini etkilediği için böyle bir yol izlenmiştir.

Kullanıcı ara yüzü olarak bir web uygulaması yazılmıştır. Kullanıcıların web sayfası üzerinde göreceği sayfalar giriş sayfası, ana sayfa ve tavsiye motoru sayfası olacaktır. Giriş sayfasında kullanıcı eğer daha önce kayıt olmuşsa bilgilerini doldurup sisteme giriş yapabileceği bir sayfa, kayıt olmamışsa ise bilgilerini girip kayıt olabileceği bileşenler olacaktır. Giriş sayfasının görüntüsü Şekil 6.2'de gösterilmektedir.



Şekil 6.2 Öneri sistemi web uygulaması giriş ekranı

Kullanıcı sisteme kayıt olduktan sonra giriş yaptıktan sonra ise bir ana sayfayla karşılaşmaktadır. Bu sayfada kullanıcıya gösterilen bir hoş geldin mesajı ve sistemin nasıl kullanılacağını gösteren bilgiler vardır. Bu sayfadan tavsiye motoru bağlantısına gidildiğinde öneri sisteminin ana işini yapan sayfa açılmaktadır. Bu sayfada kullanıcı gerekli bilgileri doldurup öneri yapılmasını istediğinde uygulamaya istek yapılır ve belli bir süre sonra bulunan öneriler kullanıcıya gösterilir ve kullanıcıya bunlardan seçim yaptırdıktan sonra veri tabanları güncellenir. Tavsiye web sayfasında yapılmış bir öneri sisteminin görüntüsü Şekil 6.3'de gösterilmektedir.



Şekil 6.3 Öneri sistemi web uygulaması tavsiye sayfası

Uygulamanın en önemli kısmı ise önerilerin hesaplandığı ve kullanıcıya bu bilgilerin gönderildiği ana uygulama ve web servisleri bölümüdür. Bu kısım genel olarak üç bileşenden oluşmaktadır. Bunlar işbirlikçi filtreleme motoru, ontoloji tabanlı durum çıkarsama motoru ve web servislerdir.

İşbirlikçi filtreleme motoru kullanıcıların bir müziği kaç kere dinlediği ile ilgili bir girdiyi kullanmaktadırlar. Bu veri "user_track" tablosundan "export" işlemiyle edilmiş bir veridir. Bu verinin örnek görüntüsü Çizelge 6.8'de gösterilmektedir.

Çizelge 6.8 Örnek CF girdisi

2,1706,2
2,2204,1
2,1232,3
2,1460,4
2,2305,4

Bu veride ilk sıradaki eleman kullanıcının tekil numarasını, ikinci sıradaki eleman kullanıcının dinlediği parçanın tekil numarasını, üçüncü sıradaki eleman ise kullanıcının bu parçayı ne kadar dinlediğini göstermektedir.

Uygulamadan gelen bilgilere göre işbirlikçi filtreleme motoru kullanıcı tabanlı veya parça tabanlı işbirlikçi filtreleme yapmaktadır. Eğer kullanıcı, kullanıcı tabanlı işbirlikçi filtreleme metodunu seçerse ve kullanıcının geçmiş işlemleri varsa bu motor seçilen algoritmayla yapılan hesaplamalardan gelen sonuçları eşik değerinin üzerindeyse gösterecektir. Eğer kullanıcının parça tabanlı işbirlikçi motorunu o zaman da kullanıcının kim olduğunun önemi yoktur. Seçilen şarkıya göre ve diğer şarkıların seçimlerine göre yakın olan sonuçları döndürecektir. İşbirlikçi filtreleme metodunun tek problemi kullanıcının geçmiş deneyimleri yoksa getirilen sonuçların ya başarım değerinin çok düşük olması ya da hiç sonuç gelmemesi olacaktır.

Uygulamadaki kullanıcı ara yüzünden gelen sisteme giriş yapmış kullanıcının demografik bilgileri ve kullanıcının müzik tercihlerine göre durum tabanlı çıkarsama motoru çalışacaktır. Kullanıcının tercihleri ilk olarak yukarıda bahsetmiş olduğumuz ontolojik modelde toplanacaktır. Daha sonra bu ontolojik model sistemden çıkarılmış olan tüm deneyimlerle karşılaştırılacaktır ve uygun olan çözümler işbirlikçi filtreleme sonuçları ile beraber birleştirilerek kullanıcıya gönderileceklerdir. Kullanıcının geçmiş deneyimlerinin tutulduğu durum tabanlı çıkarsama dosyasından alınmış örnek bir durum Çizelge 6.9'da gösterilmiştir.

Çizelge 6.9 Örnek bir durum gösterimi

```
<rdf:Description rdf:about="http://www.example.com/ontologies/recommend.owl#I648394">
   <HAS-AGE rdf:datatype="http://www.w3.org/2001/XMLSchema#string">18-24</HAS-AGE>
   <HAS-TAG rdf:datatype="http://www.w3.org/2001/XMLSchema#int">514</HAS-TAG>
   <HAS-REGISTER rdf:datatype="http://www.w3.org/2001/XMLSchema#string">2006</HAS-REGISTER>
   <HAS-GENDER rdf:datatype="http://www.w3.org/2001/XMLSchema#string">m</HAS-GENDER>
   <HAS-COUNTRY rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Romania</HAS-COUNTRY>
   <HAS-REGISTER rdf:datatype="http://www.w3.org/2001/XMLSchema#string">2007</HAS-REGISTER>
   <HAS-TAG rdf:datatype="http://www.w3.org/2001/XMLSchema#int">806</HAS-TAG>
   <HAS-TAG rdf:datatype="http://www.w3.org/2001/XMLSchema#int">227</HAS-TAG>
   <rdf:type rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE"/>
   <HAS-TAG rdf:datatype="http://www.w3.org/2001/XMLSchema#int">262</HAS-TAG>
   <HAS-GENDER rdf:datatype="http://www.w3.org/2001/XMLSchema#string">f</HAS-GENDER>
   <HAS-TAG rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1643</HAS-TAG>
   <HAS-COUNTRY rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Austria</HAS-COUNTRY>
</rdf:Description >
```

Yukarıdaki kod parçacığı öneri sisteminden alınmış bir problem ve çözüm durumunu içermektedir. Kullanıcı öneri istediğinde gönderdiği bilgiler ile yukarıdaki durumum karşılaştırılması yapılır ve eşik değeri üzerindeyse sonuç sisteme bildirilir. Burada öneri sisteminin tavsiye başarımını arttırmak ve kullanıcı memnuniyetini sağlamak açısından yaptığı diğer önemli bir katkı ise ontolojik çıkarım yapılmasıdır.

Ontolojik sonuç çıkarımı şu şekilde çalışmaktadır. Kullanıcının tercih yaparken "hard-rock" adlı müzik kategorisini seçsin. Ontoloji modeli oluşturulurken müzik kategorileri üst ve alt kategoriler şeklinde oluşturulmuştur. Ontolojik sonuç çıkarımı ise model bileşenlerinin birbiriyle ilişkisi üzerine çalışmaktadır. Örneğin "rock" kategorisi iki alt kategoriye sahiptir. Bunlar "hard rock" ve "soft rock" şeklindedir. Kullanıcı "hard rock" kategorisini tercih ettiğinde sistem ilk olarak "hard rock" kategorisine sahip parçaları bulur. Eğer bu kategori başka bir kategoriden türemiş ise de ona ait olan parçalarla da karşılaştırma yapar. Örnekte görüldüğü gibi "hard-rock" seçimini yapan bir kullanıcıya "rock" tipinden şarkılar da gösterilebilmektedir. Bu çıkarımlar sistemin başarımını arttırmakta ve kullanıcı bağımlılığını sağlamaktadır.

İki tavsiye motorundan da öneriler alındıktan sonra web servisler bu verileri işlemektedir. Uygulamada belirtilen sayıya göre ve sistemden elde edilen başarım sırasına göre bu parçalar kullanıcıya gösterilmektedir. Ancak gösterilen bu parçalar başarım sırasına göre değil, rastgele olarak gösterilmektedir. Bu şekilde kullanıcının seçimlerine göre tavsiye başarımı hesap edilmektedir.

Son olarak ise veri tabalarının güncellenmesi durumu vardır. Kullanıcıya gösterilen parçalardan sonra kullanıcıdan alınan geri bildirimler ışığında işbirlikçi filtreleme veri abanı ve durum tabanlı çıkarsama veri tabanı güncellenir.

BÖLÜM 7

DENEY SONUÇLARI ve DEĞERLENDİRME

Filtrelenmiş verilerden elde edilen sonuçlar bu kısımda listelenecektir. Deneysel sonuçlar üç ana grupta toplanmaktadır. Bunlar Kullanıcı-Etiket verisi üzerinden elde edilen sonuçlar, sadece İngiltere (United-Kingdom) ülkesine ait olan kullanıcılardan elde edilen Kullanıcı – Şarkı verisine ait sonuçlar ve tüm kullanıcılardan elde edilen Kullanıcı - Şarkı verisine ait sonuçlardır. Her grup için test işlemleri 10 önerim ve 20 önerim yapıldığı durumlar üzerinden değerlendirilmektedir. Her grup için aşağıdaki algoritmalar kullanılacaktır.

- İşbirlikçi filtreleme
- İşbirlikçi filtreleme ve durum tabanlı çıkarsama
- İşbirlikçi filtreleme ve ontolojik çıkarımlı durum tabanlı çıkarsama

Her önerim grubu ise eldeki veriden %10'u, %50'si ve %90'ı eğitim verisi olarak kullanılarak üç ayrı durum değerlendirilmesi yapılmıştır. Bu ayrımın amacı eldeki veri az olduğu veya çok olduğu durumlarda durum tabanlı çıkarsamanın önerimlere katkısı olup olmadığını görmek olacaktır. Ayrıca yapılan her test farklı komşuluk değerleri ile de değerlendirilerek faklı komşuluk durumlarında önerilen metodolojinin nasıl davrandığı ölçülmektedir. Yapılan her test için precision, recall ve f-measure grafikleri listelenecektir.

7.1 Deneysel Sonuçlar

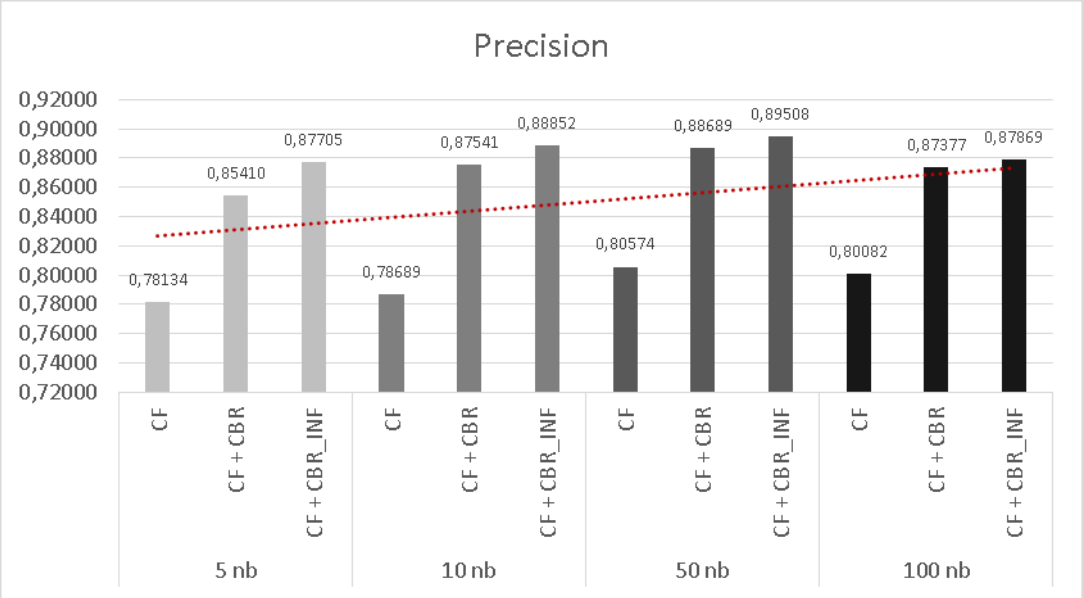
Bu bölümde kullanıcı-etiket, kullanıcı-şarkı (tüm ülkeler), kullanıcı-şarkı (tek ülke) testlerinin sonuçları listelenecektir.

7.1.1 Kullanıcı - Etiket Öneri Sonuçları

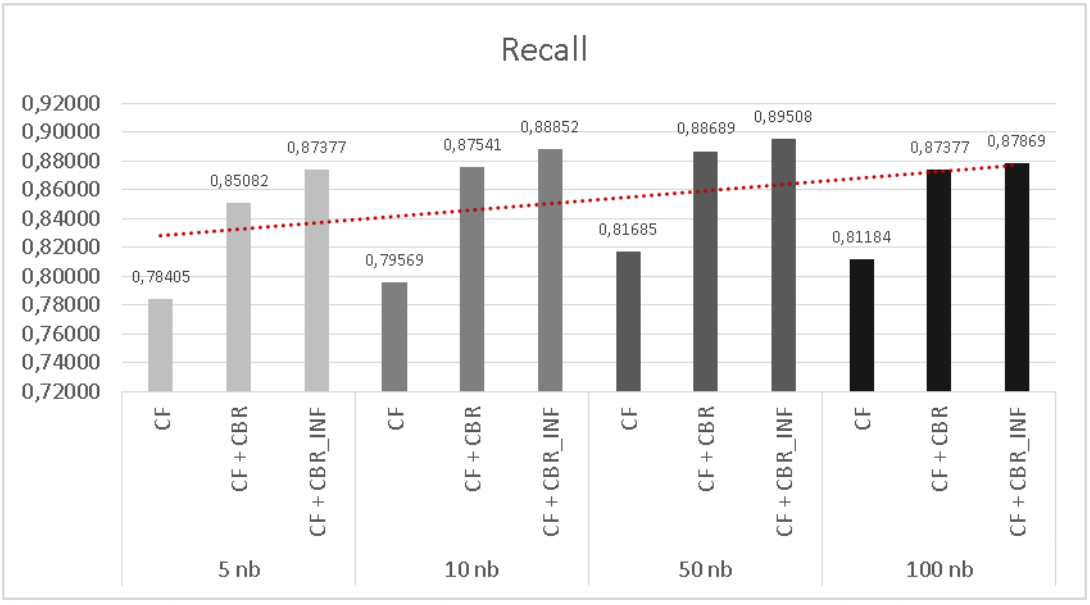
Kullanıcı – Etiket testlerinin değerlendirilmesi bu bölümde incelenecektir. 10 önerim ve 20 önerimli testler farklı komşuluk değerleriyle hesaplanarak grafik gösterimi yapılacaktır.

7.1.1.1 10 Öneri Sonuçları (%10 Eğitim Verisi)

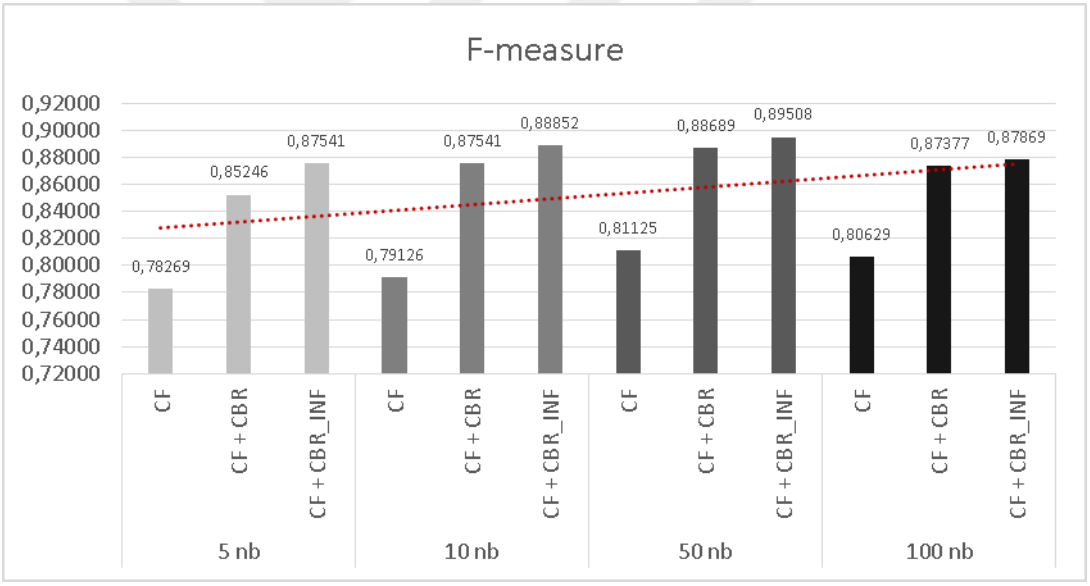
Şekil 7.1, Şekil 7.2, Şekil 7.3 tüm kullanıcı-etiket verisinin %10' u kullanılarak elde edilen 10 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.1 Kullanıcı - Etiket verisi (%10 eğitim - 10 öneri) precision grafiği



Şekil 7.2 Kullanıcı - Etiket verisi (%10 eğitim - 10 öneri) recall grafiği

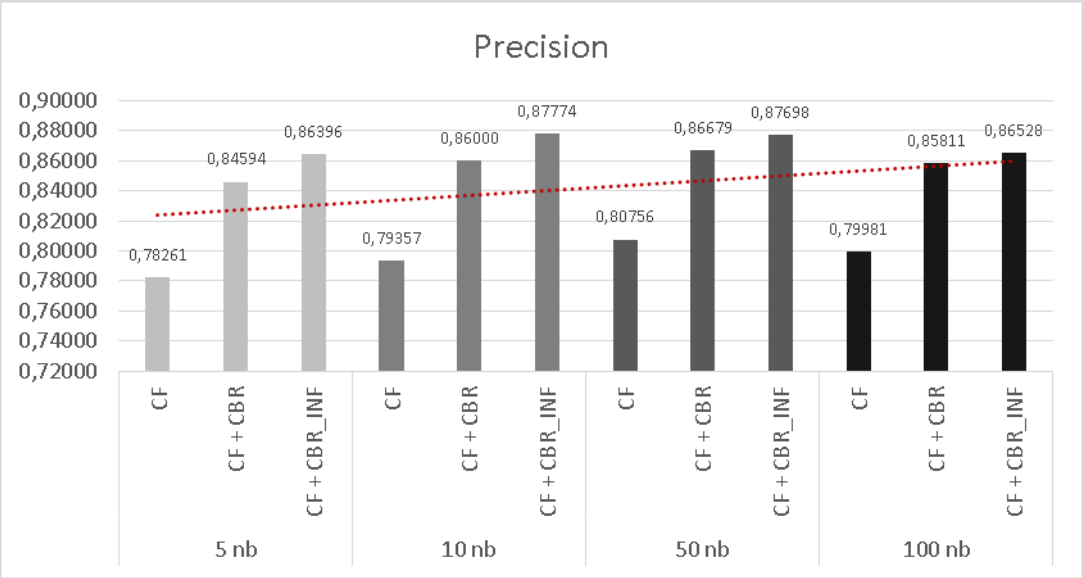


Şekil 7.3 Kullanıcı - Etiket verisi (%10 eğitim - 10 öneri) f-measure grafiği

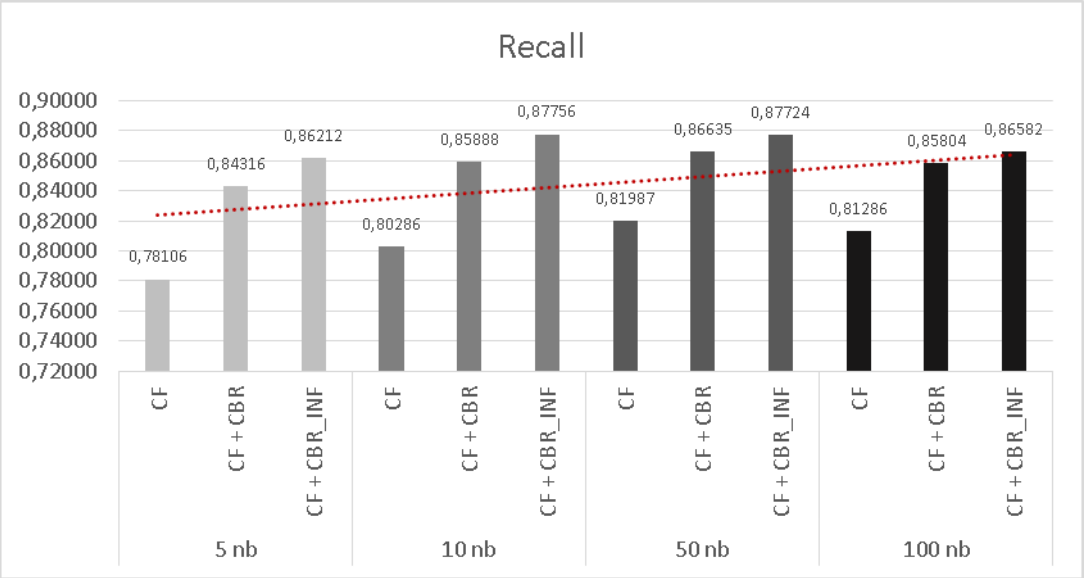
Kullanıcı - Etiket verisinin % 10'u kullanılarak 10 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda hep artış gösterdiği izlenmektedir. Yapılan testler 5, 10, 50 ve 100 komşuluk için tekrarlanmış ve artış durumun değişmediği gözlemlenmektedir.

7.1.1.2 10 Öneri Sonuçları (%50 Eğitim Verisi)

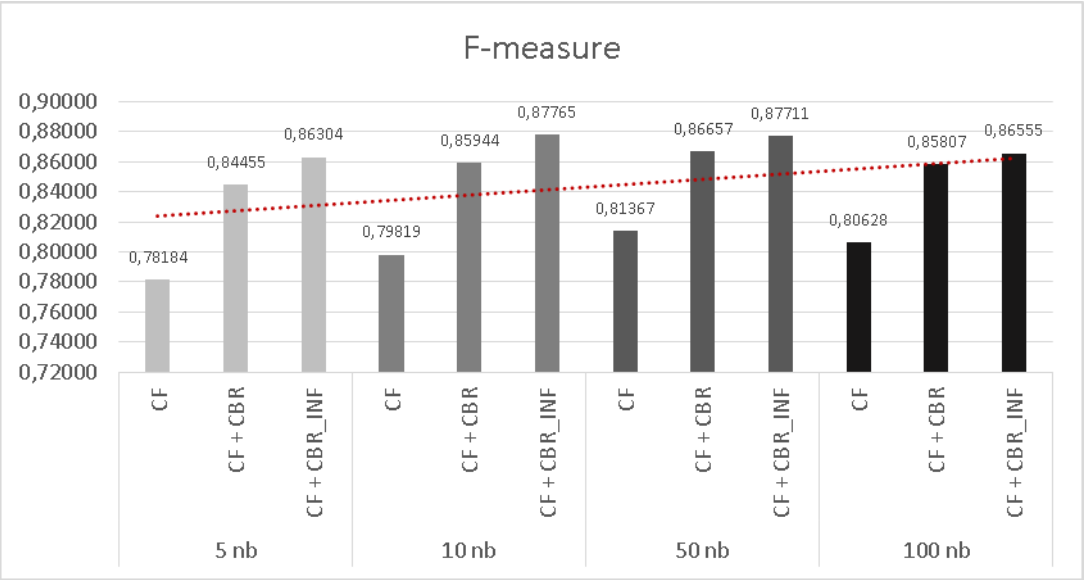
Şekil 7.4, Şekil 7.5, Şekil 7.6 tüm kullanıcı-etiket verisinin %50' si kullanılarak elde edilen 10 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.4 Kullanıcı - Etiket verisi (%50 eğitim - 10 öneri) precision grafiği



Şekil 7.5 Kullanıcı - Etiket verisi (%50 eğitim - 10 öneri) recall grafiği

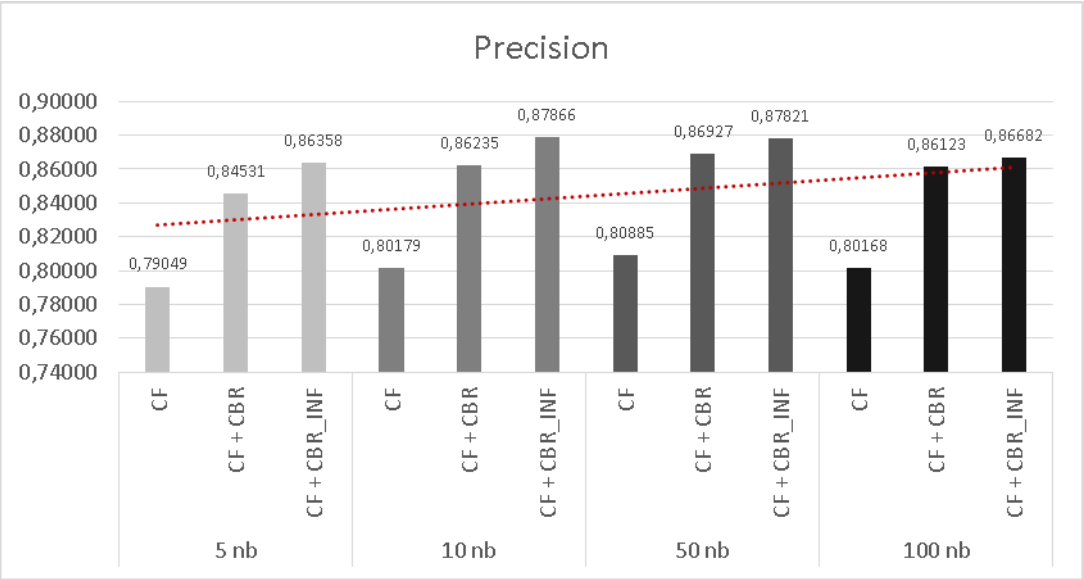


Şekil 7.6 Kullanıcı - Etiket verisi (%50 eğitim - 10 öneri) f-measure grafiği

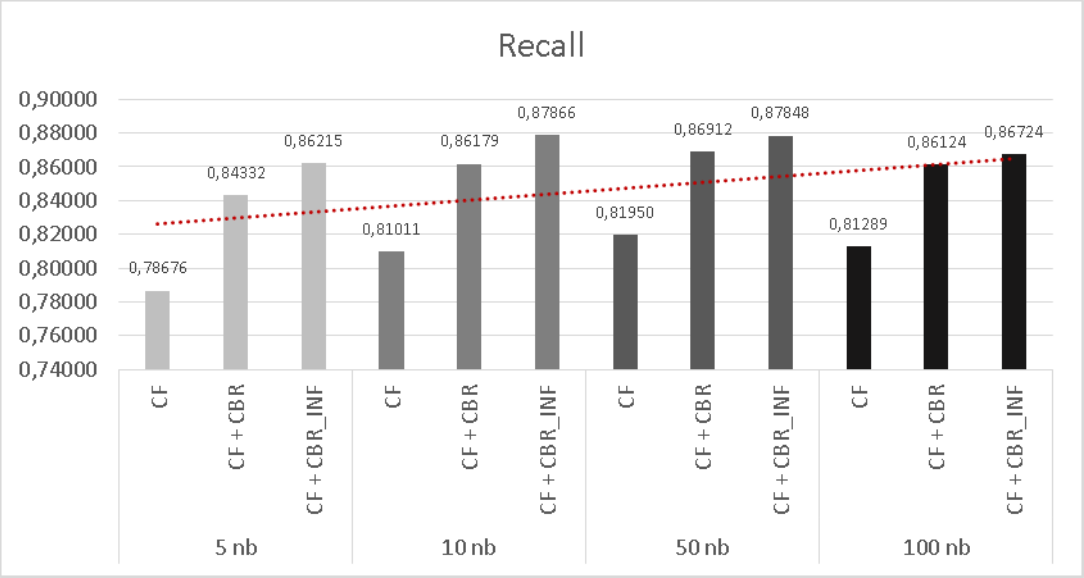
Kullanıcı - Etiket verisinin % 50'si kullanılarak 10 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda hep artış gösterdiği izlenmektedir. Yapılan testler 5, 10, 50 ve 100 komşuluk için tekrarlanmış ve artış durumunun değişmediği gözlemlenmektedir.

7.1.1.3 10 Öneri Sonuçları (%90 Eğitim Verisi)

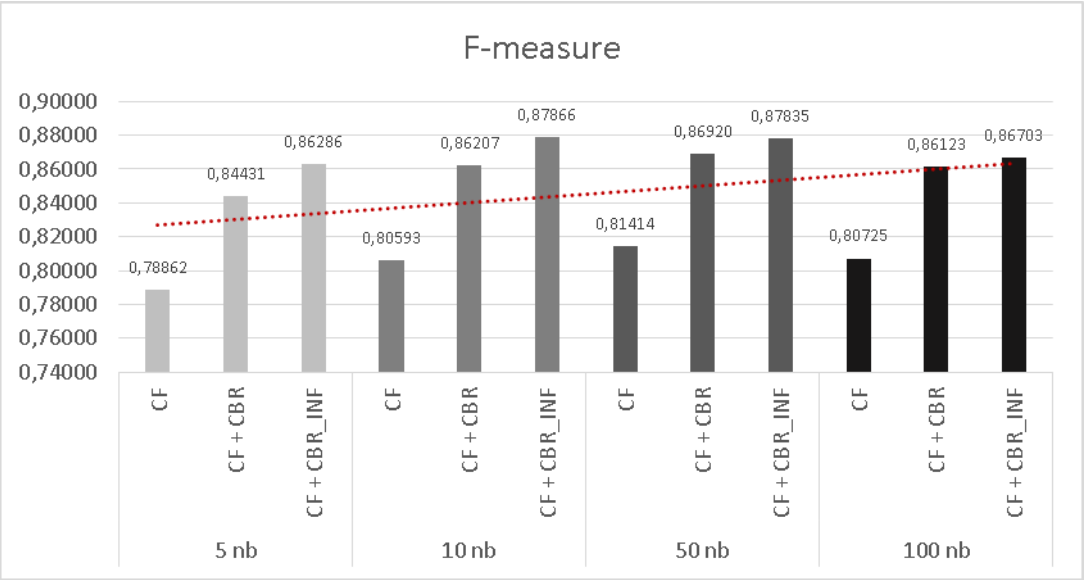
Şekil 7.7, Şekil 7.8, Şekil 7.9 tüm kullanıcı-etiket verisinin %90' ı kullanılarak elde edilen 10 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.7 Kullanıcı - Etiket verisi (%90 eğitim - 10 öneri) precision grafiği



Şekil 7.8 Kullanıcı - Etiket verisi (%90 eğitim - 10 öneri) recall grafiği

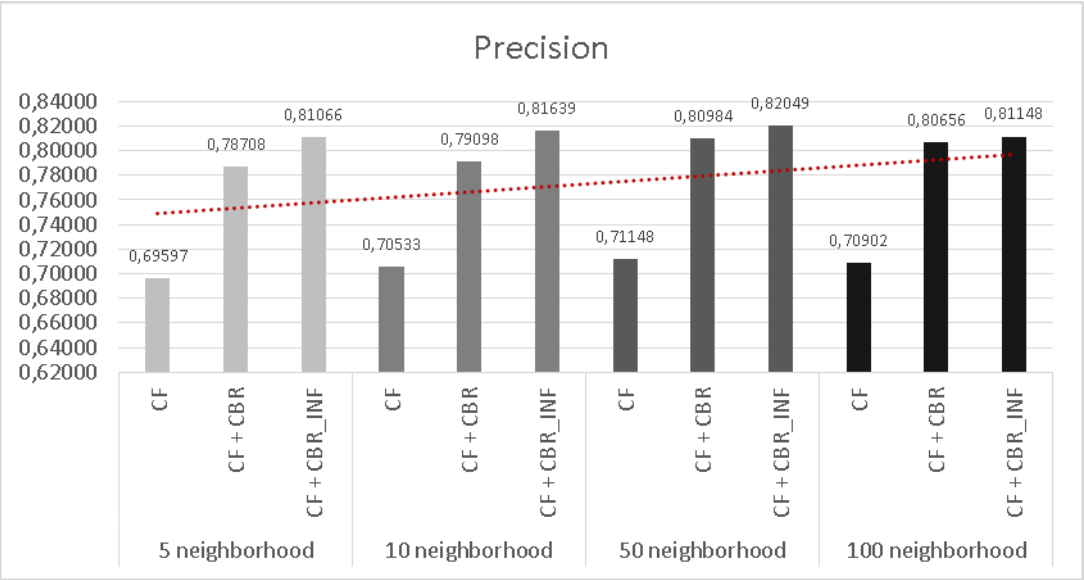


Şekil 7.9 Kullanıcı - Etiket verisi (%90 eğitim - 10 öneri) f-measure grafiği

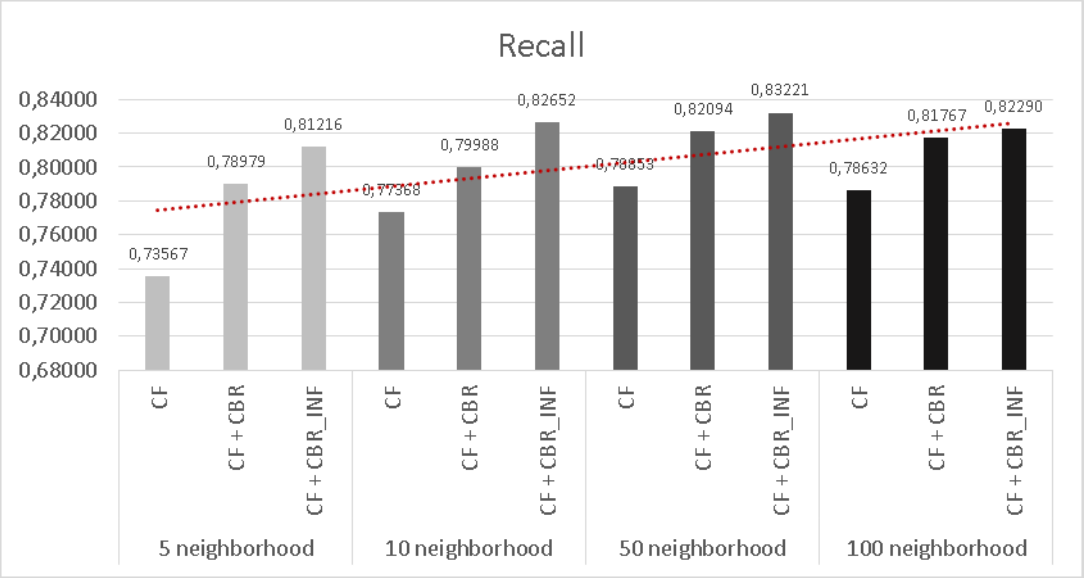
Kullanıcı - Etiket verisinin % 90'ı kullanılarak 10 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda hep artış gösterdiği izlenmektedir. Yapılan testler 5, 10, 50 ve 100 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir.

7.1.1.4 20 Öneri Sonuçları (%10 Eğitim Verisi)

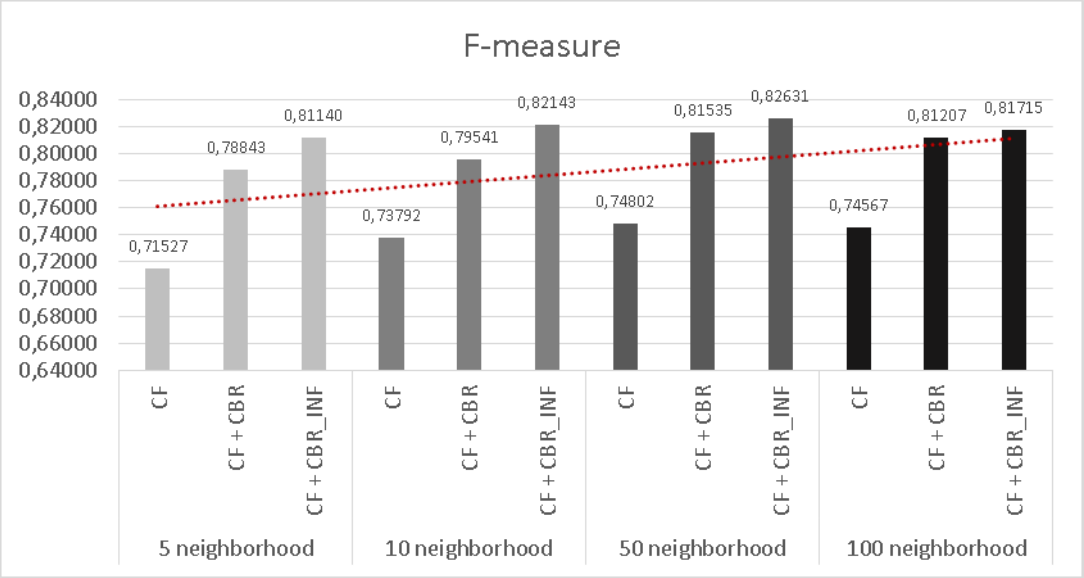
Şekil 7.10, Şekil 7.11, Şekil 7.12 tüm kullanıcı-etiket verisinin %10' u kullanılarak elde edilen 20 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.10 Kullanıcı - Etiket verisi (%10 eğitim - 20 öneri) precision grafiği



Şekil 7.11 Kullanıcı - Etiket verisi (%10 eğitim - 20 öneri) recall grafiği

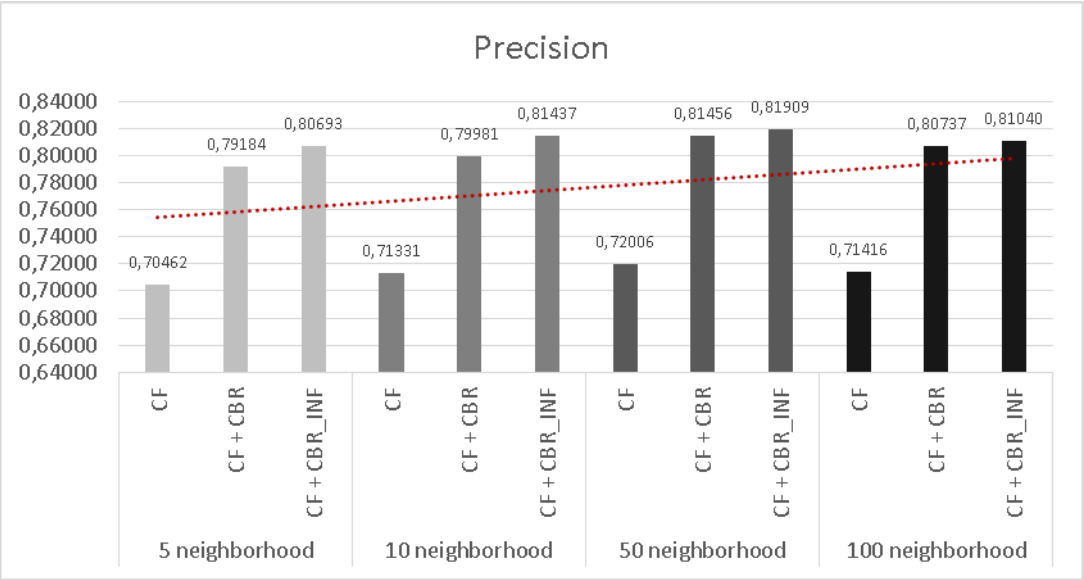


Şekil 7.12 Kullanıcı - Etiket verisi (%10 eğitim - 20 öneri) f-measure grafiği

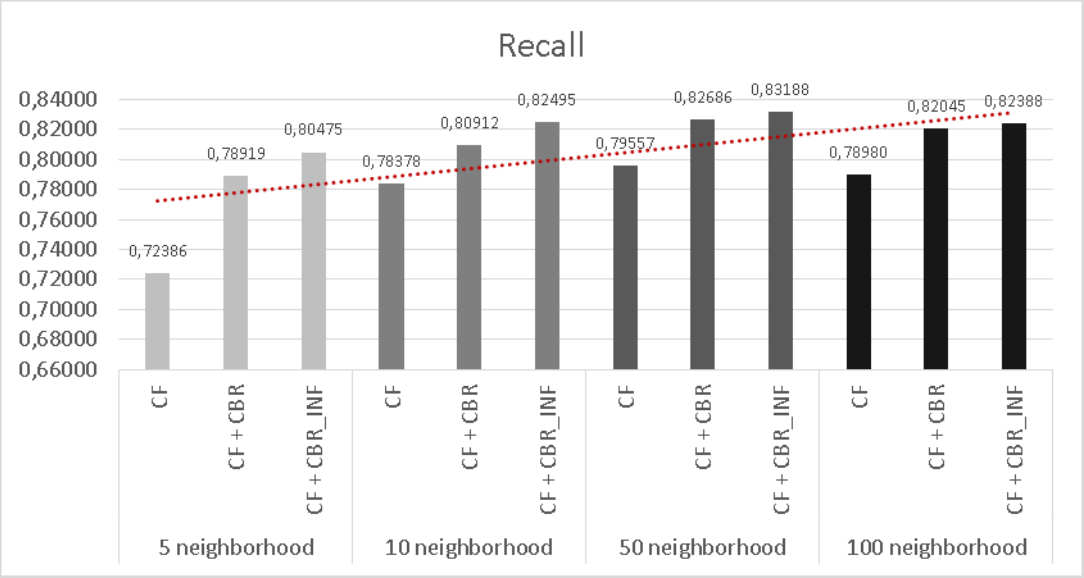
Kullanıcı - Etiket verisinin % 10'u kullanılarak 20 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda hep artış gösterdiği izlenmektedir. Yapılan testler 5, 10, 50 ve 100 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir.

7.1.1.5 20 Öneri Sonuçları (%50 Eğitim Verisi)

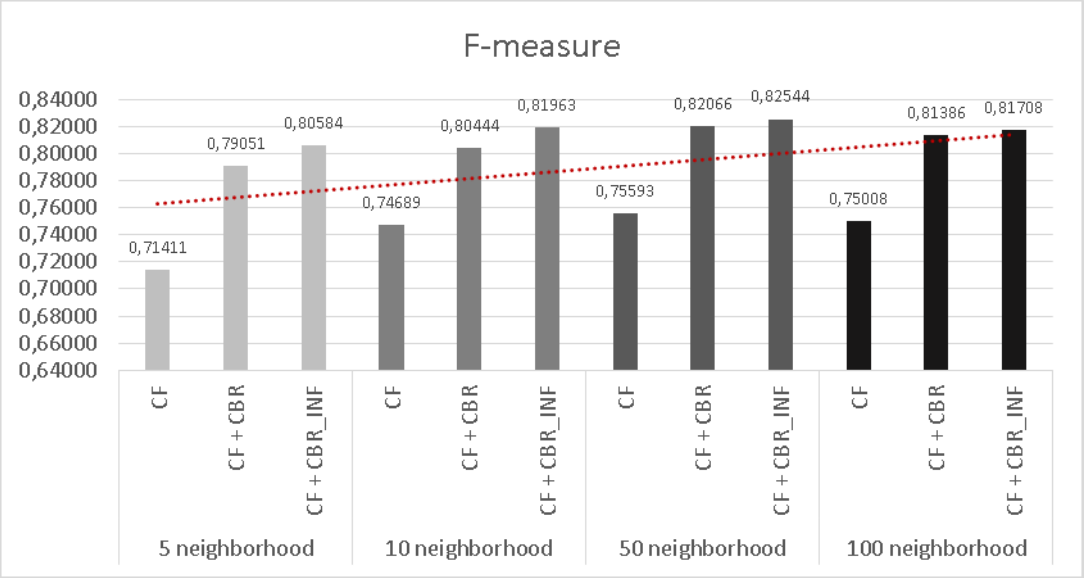
Şekil 7.13, Şekil 7.14, Şekil 7.15 tüm kullanıcı-etiket verisinin %50' si kullanılarak elde edilen 20 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.13 Kullanıcı - Etiket verisi (%50 eğitim - 20 öneri) precision grafiği



Şekil 7.14 Kullanıcı - Etiket verisi (%50 eğitim - 20 öneri) recall grafiği

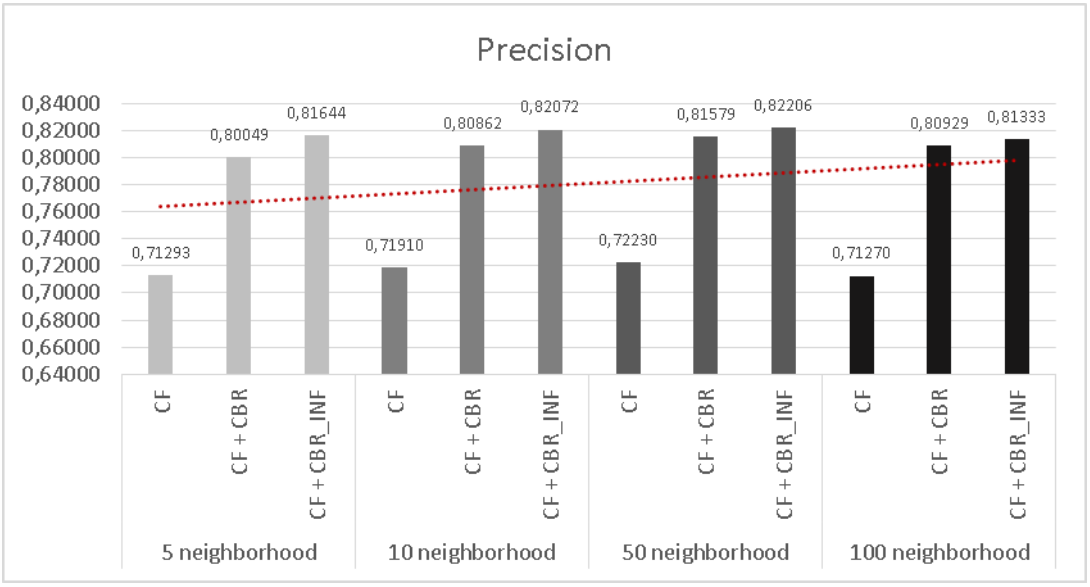


Şekil 7.15 Kullanıcı - Etiket verisi (%50 eğitim - 20 öneri) f-measure grafiği

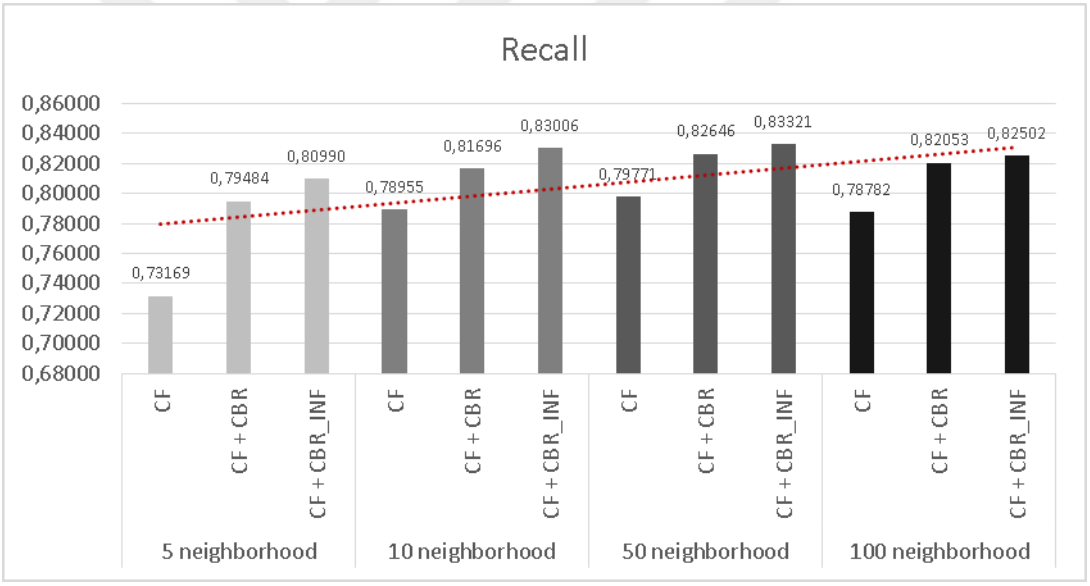
Kullanıcı - Etiket verisinin % 50'si kullanılarak 20 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda hep artış gösterdiği izlenmektedir. Yapılan testler 5, 10, 50 ve 100 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir.

7.1.1.6 20 Öneri Sonuçları (%90 Eğitim Verisi)

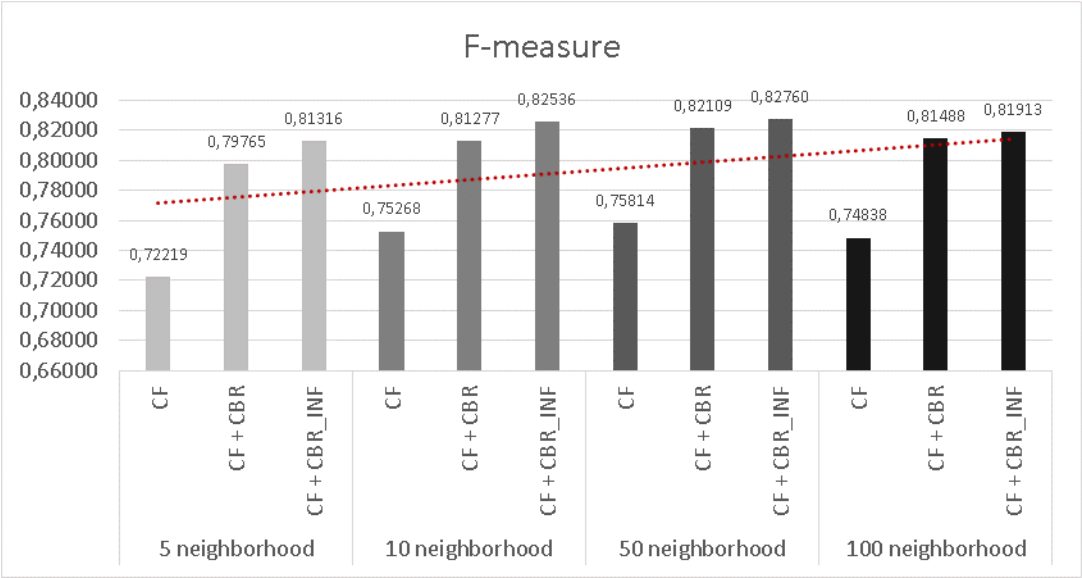
Şekil 7.16, Şekil 7.17, Şekil 7.18 tüm kullanıcı-etiket verisinin %90' ı kullanılarak elde edilen 20 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.16 Kullanıcı - Etiket verisi (%90 eğitim - 20 öneri) precision grafiği



Şekil 7.17 Kullanıcı - Etiket verisi (%90 eğitim - 20 öneri) recall grafiği



Şekil 7.18 Kullanıcı - Etiket verisi (%90 eğitim - 20 öneri) f-measure grafiği

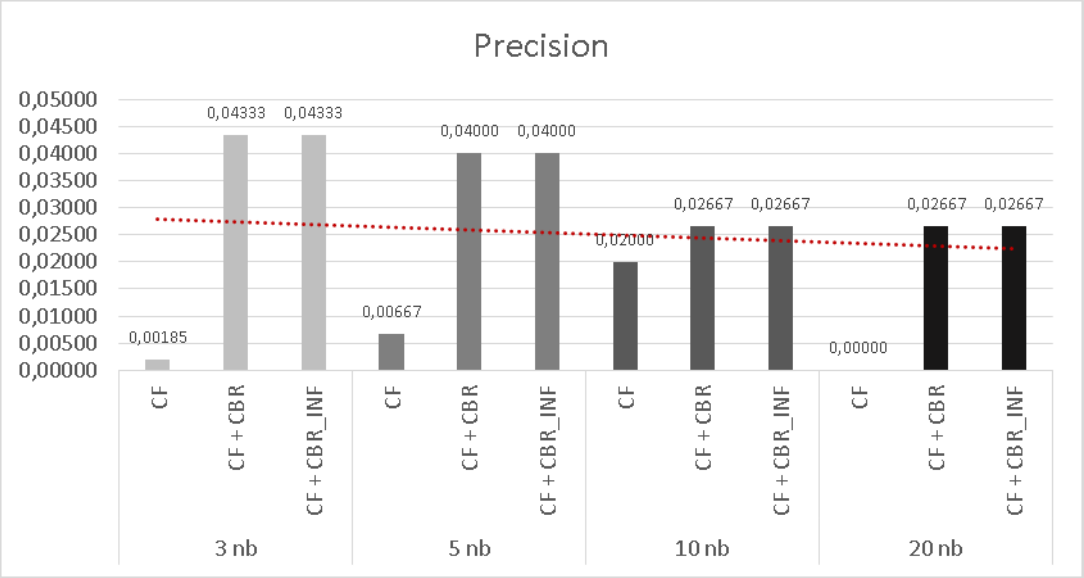
Kullanıcı - Etiket verisinin % 90'ı kullanılarak 20 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda hep artış gösterdiği izlenmektedir. Yapılan testler 5, 10, 50 ve 100 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir.

7.1.2 Kullanıcı - Şarkı Öneri Sonuçları (Tek Ülke - İngiltere)

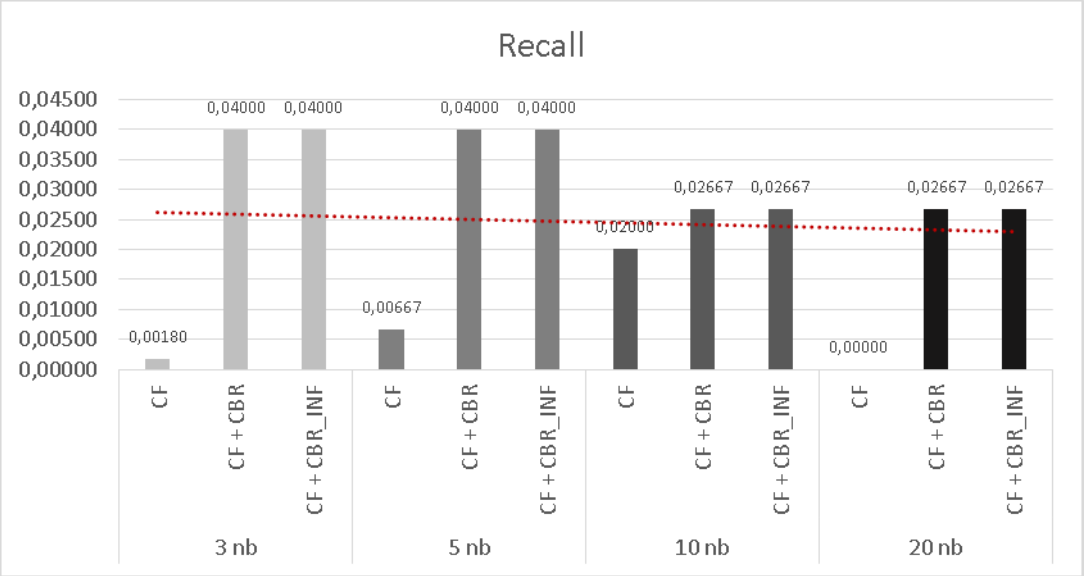
Kullanıcı – Şarkı (tek ülke için) testlerinin değerlendirilmesi bu bölümde incelenecektir. 10 önerim ve 20 önerimli testler farklı komşuluk değerleriyle hesaplanarak grafik gösterimi yapılacaktır.

7.1.2.1 10 Öneri Sonuçları (%10 Eğitim Verisi)

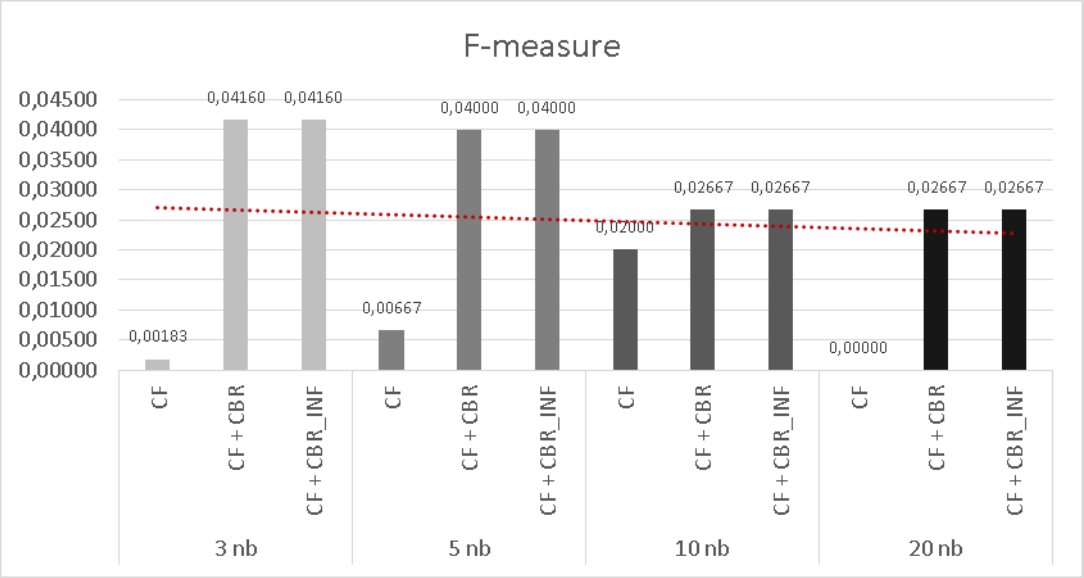
Şekil 7.19, Şekil 7.20, Şekil 7.21 tek ülke kullanıcı-şarkı verisinin %10' u kullanılarak elde edilen 10 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.19 Kullanıcı - Şarkı verisi tek ülke (%10 eğitim - 10 öneri) precision grafiği



Şekil 7.20 Kullanıcı - Şarkı verisi tek ülke (%10 eğitim - 10 öneri) recall grafiği

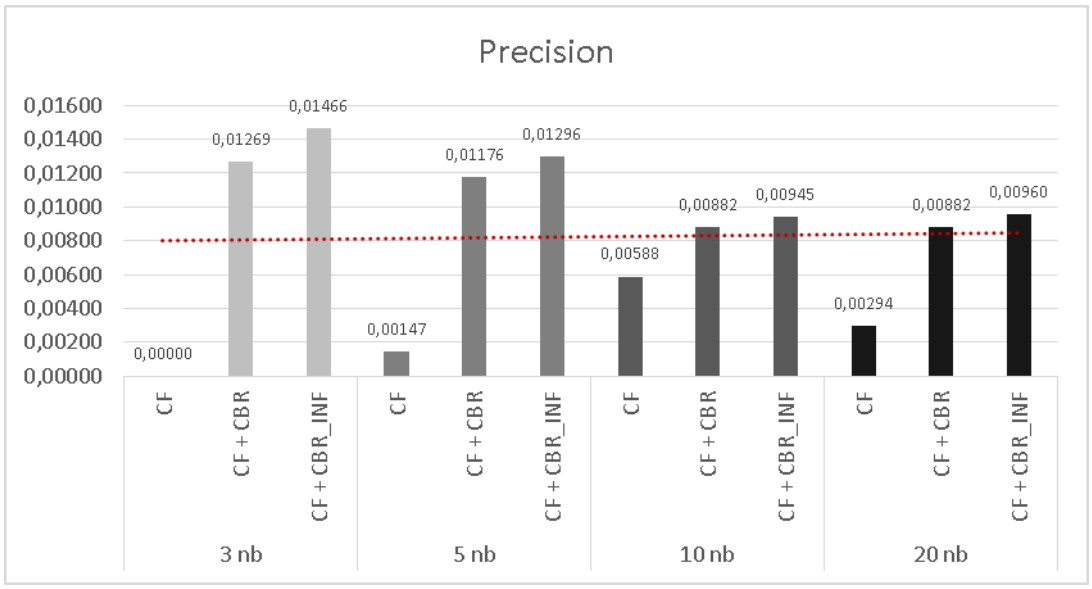


Şekil 7.21 Kullanıcı - Şarkı verisi tek ülke (%10 eğitim - 10 öneri) f-measure grafiği

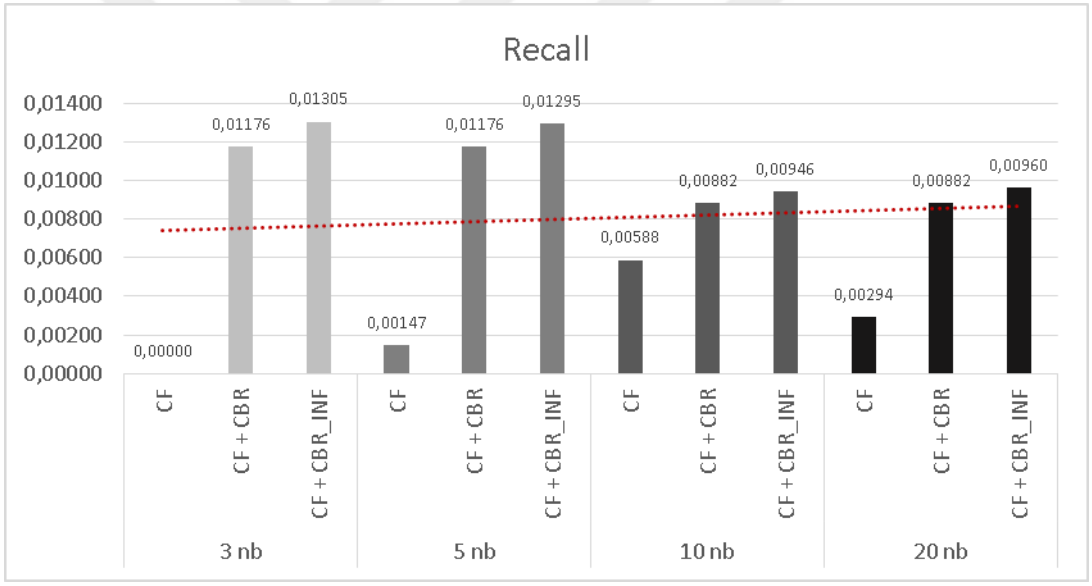
Tek ülke için Kullanıcı - Şarkı verisinin % 10'u kullanılarak 10 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda artış gösterdiği izlenmektedir. Ancak ontolojik çıkarımlı durum tabanlı çıkarsama metodu sonuçları değiştirememektedir. Yapılan testler 3, 5, 10 ve 20 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir. Komşuluk değeri arttığı durumlarda değerlerde gözlenen düşüsün sebebi ise kullanıcı sayısının az, şarkı çeşitliliğinin fazla olmasından kaynaklanmaktadır.

7.1.2.2 10 Öneri Sonuçları (%50 Eğitim Verisi)

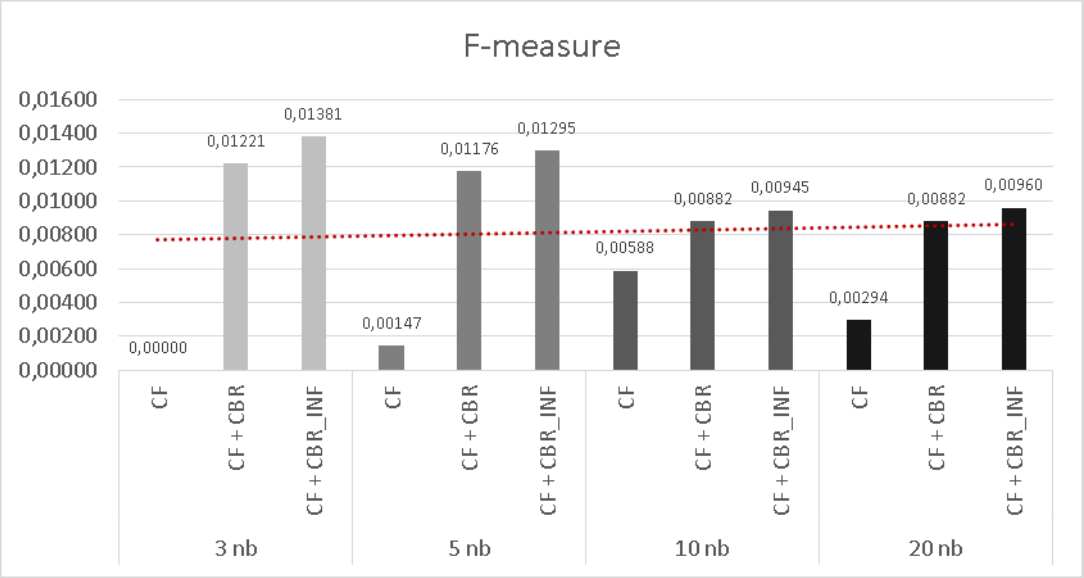
Şekil 7.22, Şekil 7.23, Şekil 7.24 tek ülke kullanıcı-şarkı verisinin %50' si kullanılarak elde edilen 10 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.22 Kullanıcı - Şarkı verisi tek ülke (%50 eğitim - 10 öneri) precision grafiği



Şekil 7.23 Kullanıcı - Şarkı verisi tek ülke (%50 eğitim - 10 öneri) recall grafiği

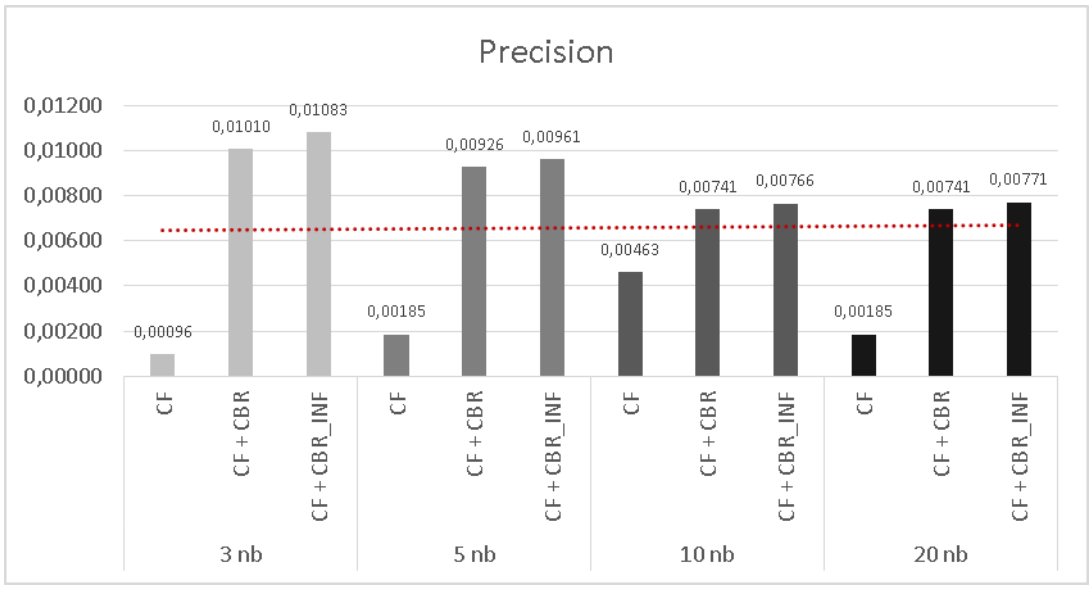


Şekil 7.24 Kullanıcı - Şarkı verisi tek ülke (%50 eğitim - 10 öneri) f-measure grafiği

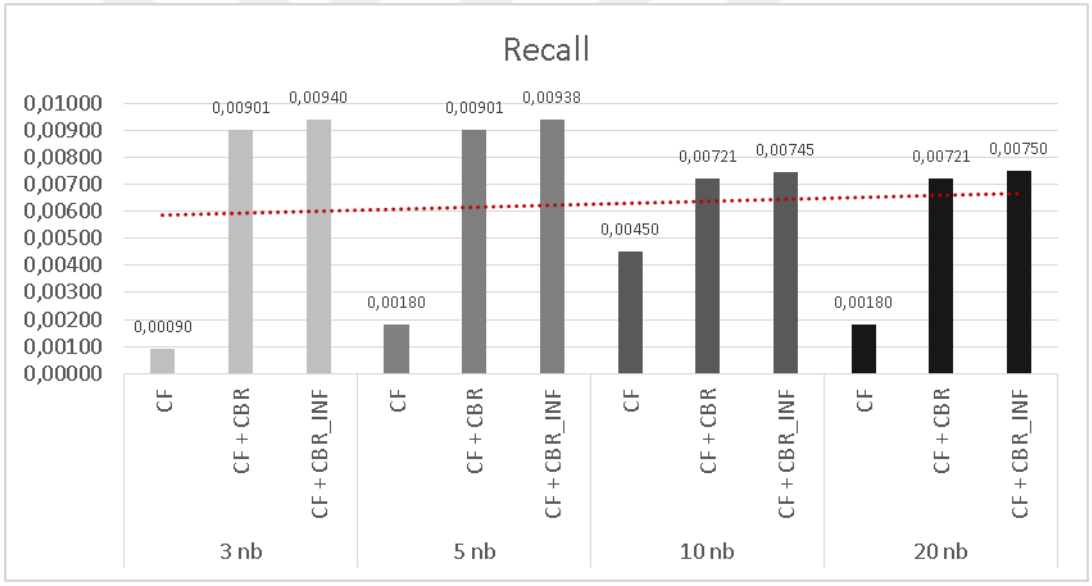
Tek ülke için Kullanıcı - Şarkı verisinin % 50'si kullanılarak 10 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda artış gösterdiği izlenmektedir. Yapılan testler 3, 5, 10 ve 20 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir.

7.1.2.3 10 Öneri Sonuçları (%90 Eğitim Verisi)

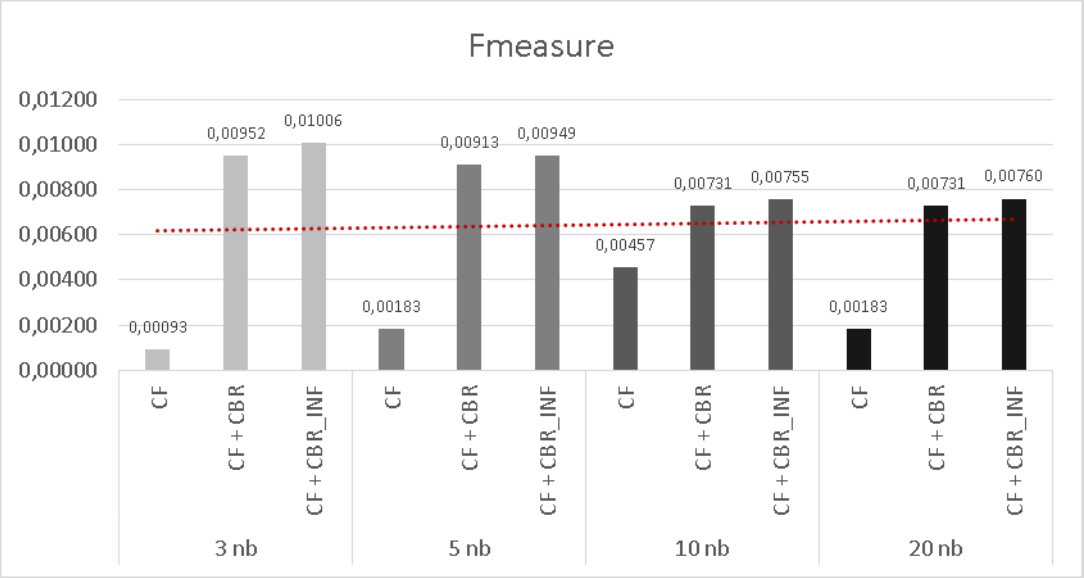
Şekil 7.25, Şekil 7.26, Şekil 7.27 tek ülke kullanıcı-şarkı verisinin %90' ı kullanılarak elde edilen 10 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.25 Kullanıcı - Şarkı verisi tek ülke (%90 eğitim - 10 öneri) precision grafiği



Şekil 7.26 Kullanıcı - Şarkı verisi tek ülke (%90 eğitim - 10 öneri) recall grafiği

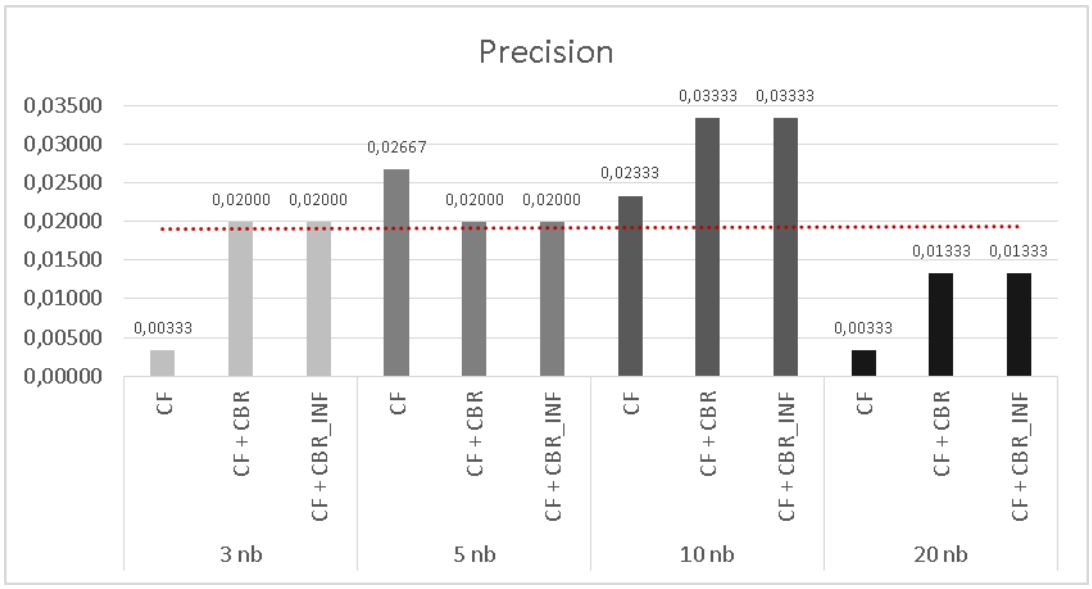


Şekil 7.27 Kullanıcı - Şarkı verisi tek ülke (%90 eğitim - 10 öneri) f-measure grafiği

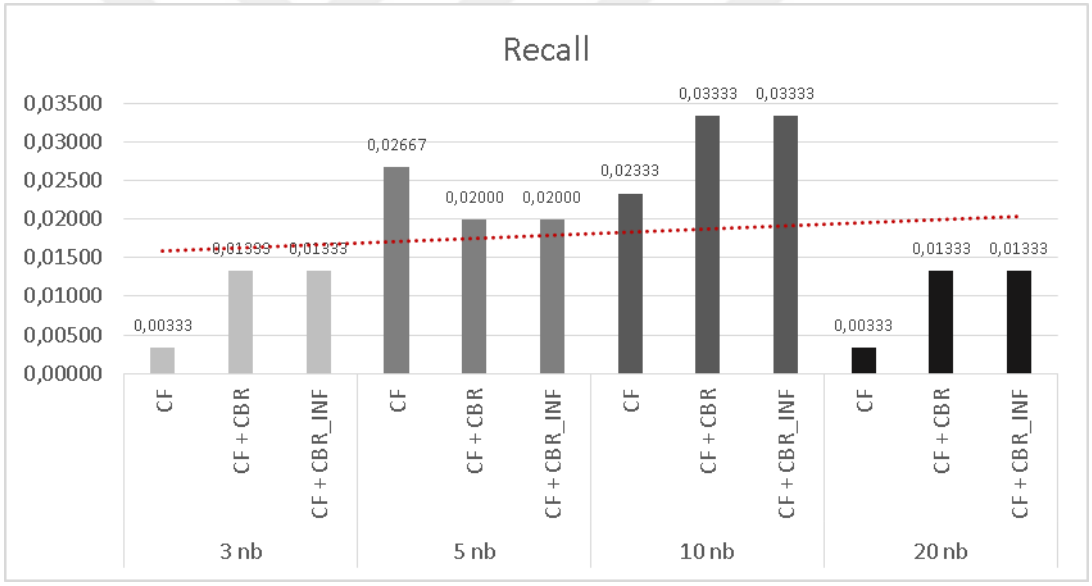
Tek ülke için Kullanıcı - Şarkı verisinin % 90'ı kullanılarak 10 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda artış gösterdiği izlenmektedir. Yapılan testler 3, 5, 10 ve 20 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir. Elde edilen değerlerin düşük olmasının sebebi veri çeşitliliğin çok fazla olmasından kaynaklanmaktadır.

7.1.2.4 20 Öneri Sonuçları (%10 Eğitim Verisi)

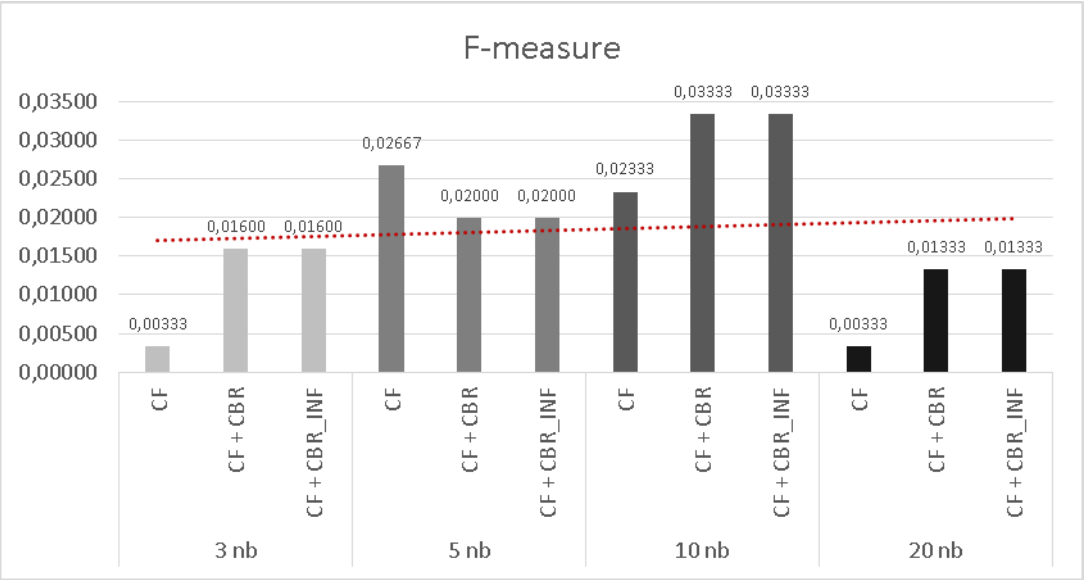
Şekil 7.28, Şekil 7.29 Şekil 7.30 tek ülke kullanıcı-şarkı verisinin %10' u kullanılarak elde edilen 20 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.28 Kullanıcı - Şarkı verisi tek ülke (%10 eğitim - 20 öneri) precision grafiği



Şekil 7.29 Kullanıcı - Şarkı verisi tek ülke (%10 eğitim - 20 öneri) recall grafiği

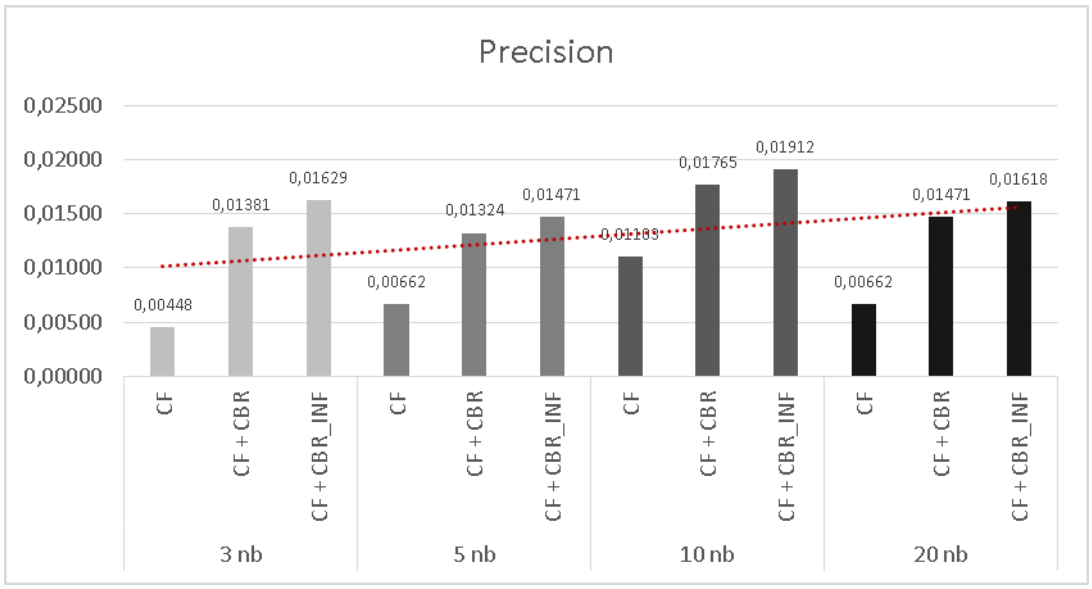


Şekil 7.30 Kullanıcı - Şarkı verisi tek ülke (%10 eğitim - 20 öneri) f-measure grafiği

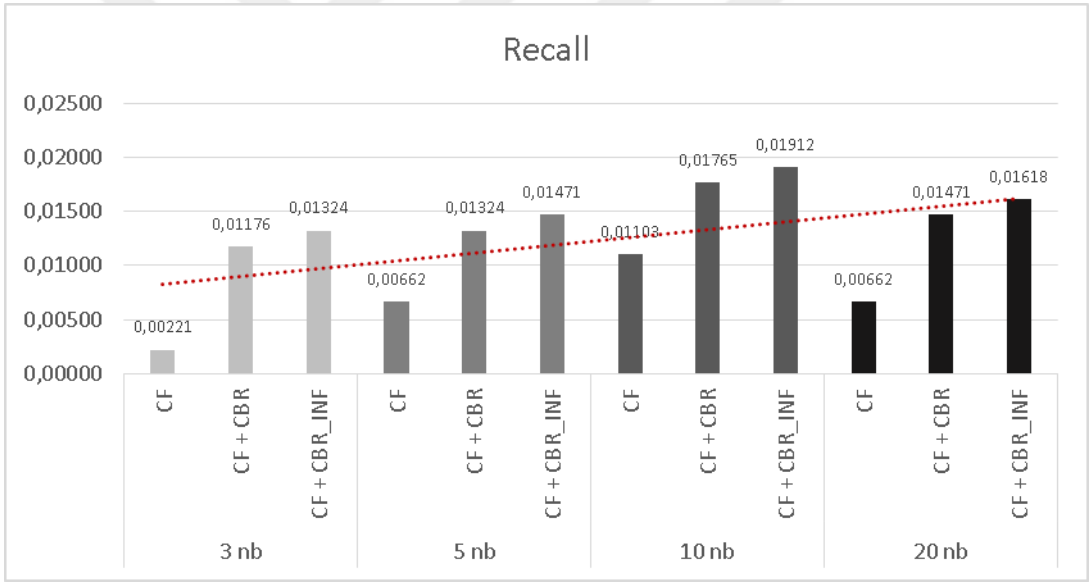
Tek ülke için Kullanıcı - Şarkı verisinin % 10'u kullanılarak 20 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda artış gösterdiği izlenmektedir. Ancak ontolojik çıkarımlı durum tabanlı çıkarsama metodu sonuçları değiştirememektedir. Yapılan testler 3, 5, 10 ve 20 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir. Elde edilen değerlerin düşük olmasının sebebi veri çeşitliliğin çok fazla olmasından kaynaklanmaktadır.

7.1.2.5 20 Öneri Sonuçları (%50 Eğitim Verisi)

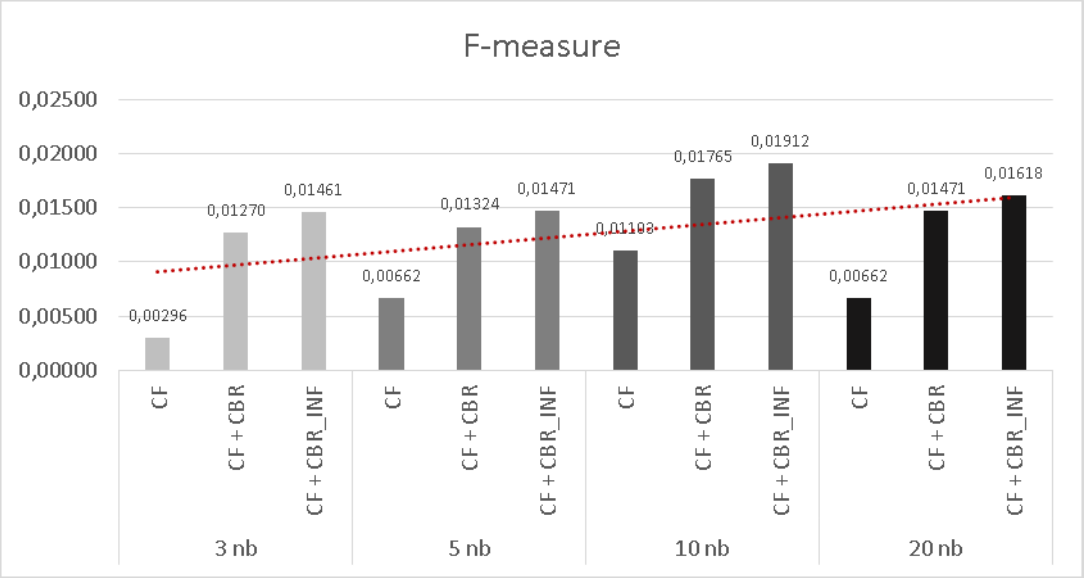
Şekil 7.31, Şekil 7.32, Şekil 7.33 tek ülke kullanıcı-şarkı verisinin %50' si kullanılarak elde edilen 20 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.31 Kullanıcı - Şarkı verisi tek ülke (%50 eğitim - 20 öneri) precision grafiği



Şekil 7.32 Kullanıcı - Şarkı verisi tek ülke (%50 eğitim - 20 öneri) recall grafiği

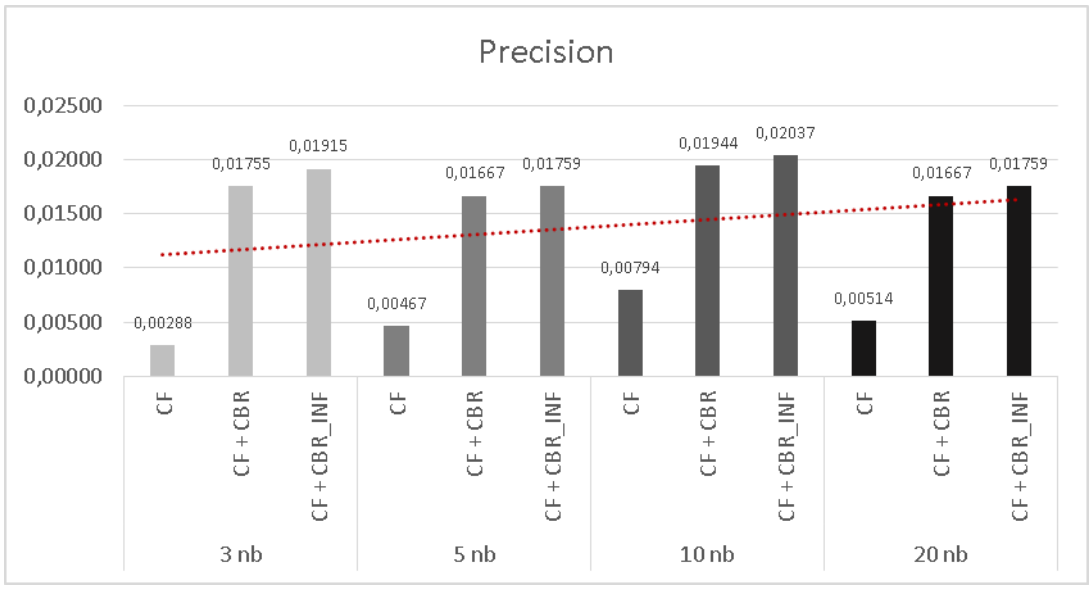


Şekil 7.33 Kullanıcı - Şarkı verisi tek ülke (%50 eğitim - 20 öneri) f-measure grafiği

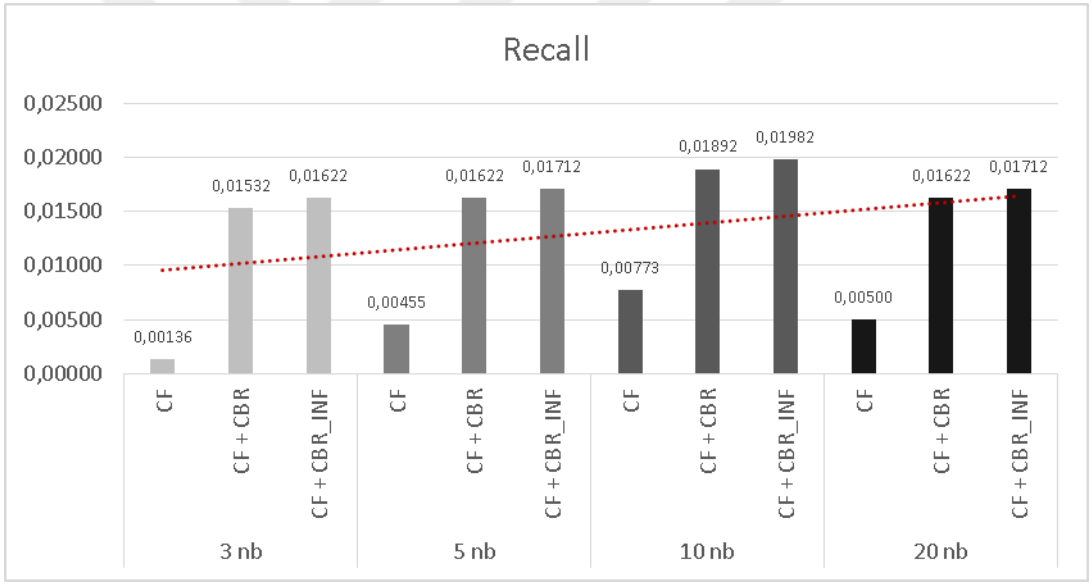
Tek ülke için Kullanıcı - Şarkı verisinin % 50'si kullanılarak 20 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda artış gösterdiği izlenmektedir. Yapılan testler 3, 5, 10 ve 20 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir. Elde edilen değerlerin düşük olmasının sebebi veri çeşitliliğin çok fazla olmasından kaynaklanmaktadır.

7.1.2.6 20 Öneri Sonuçları (%90 Eğitim Verisi)

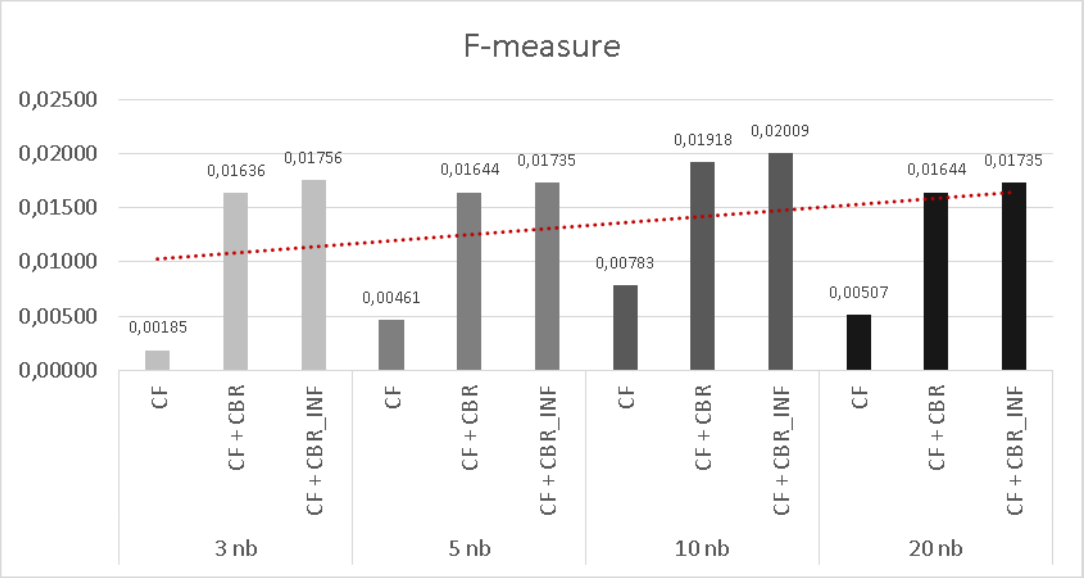
Şekil 7.34, Şekil 7.35, Şekil 7.36 tek ülke kullanıcı-şarkı verisinin %90' ı kullanılarak elde edilen 20 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.34 Kullanıcı - Şarkı verisi tek ülke (%90 eğitim - 20 öneri) precision grafiği



Şekil 7.35 Kullanıcı - Şarkı verisi tek ülke (%90 eğitim - 20 öneri) recall grafiği



Şekil 7.36 Kullanıcı - Şarkı verisi tek ülke (%90 eğitim - 20 öneri) f-measure grafiği

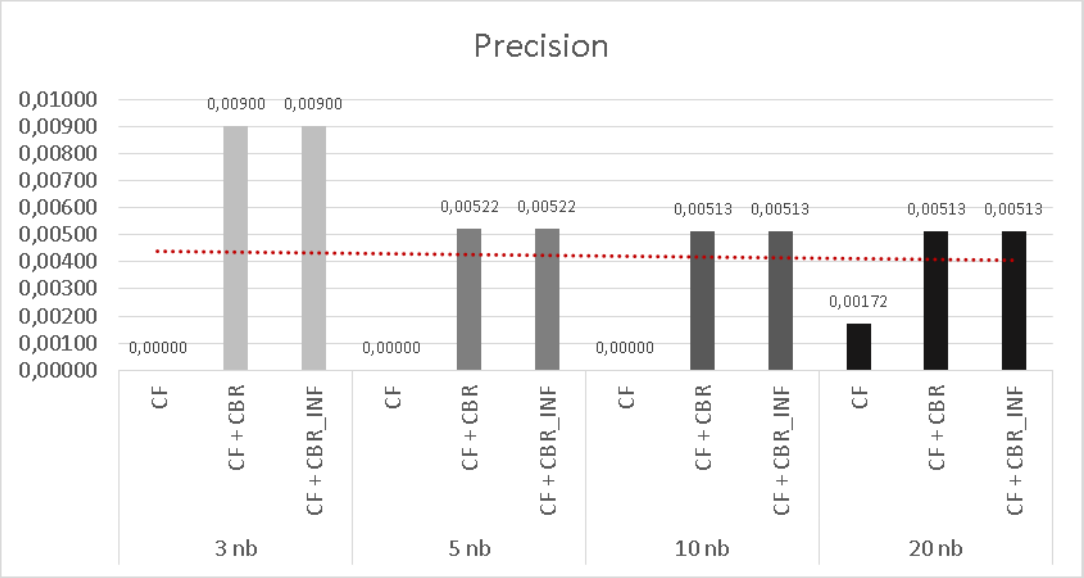
Tek ülke için Kullanıcı - Şarkı verisinin % 90'ı kullanılarak 20 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda artış gösterdiği izlenmektedir. Yapılan testler 3, 5, 10 ve 20 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir. Elde edilen değerlerin düşük olmasının sebebi veri çeşitliliğin çok fazla olmasından kaynaklanmaktadır.

7.1.3 Tüm Şarkılar Üzerinden Yapılan Öneri Sonuçları

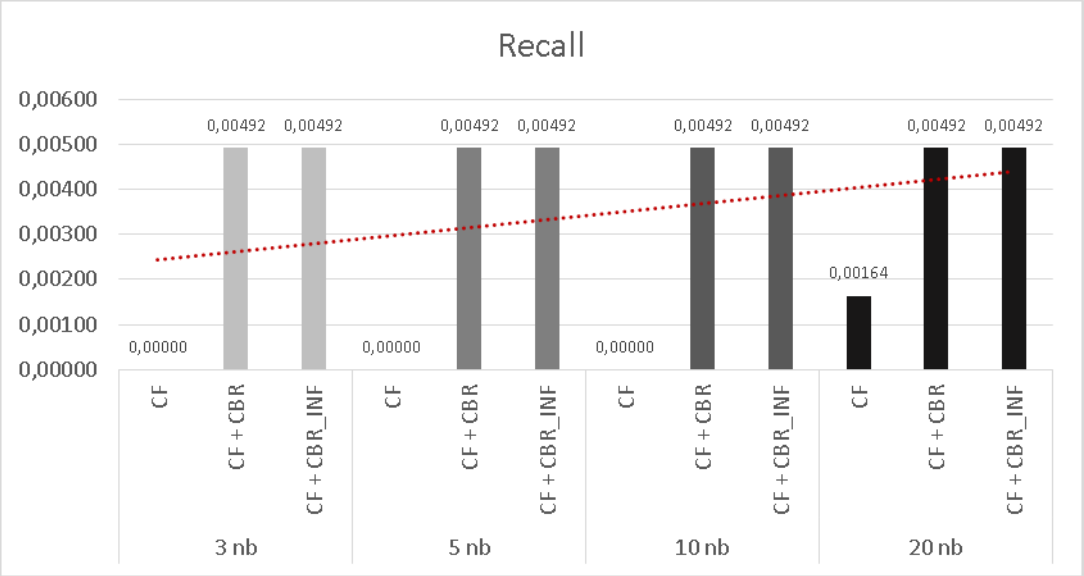
Kullanıcı – Şarkı (tüm ülkeler için) testlerinin değerlendirilmesi bu bölümde incelenecektir. 10 önerim ve 20 önerimli testler farklı komşuluk değerleriyle hesaplanarak grafik gösterimi yapılacaktır.

7.1.3.1 10 Öneri Sonuçları (%10 Eğitim Verisi)

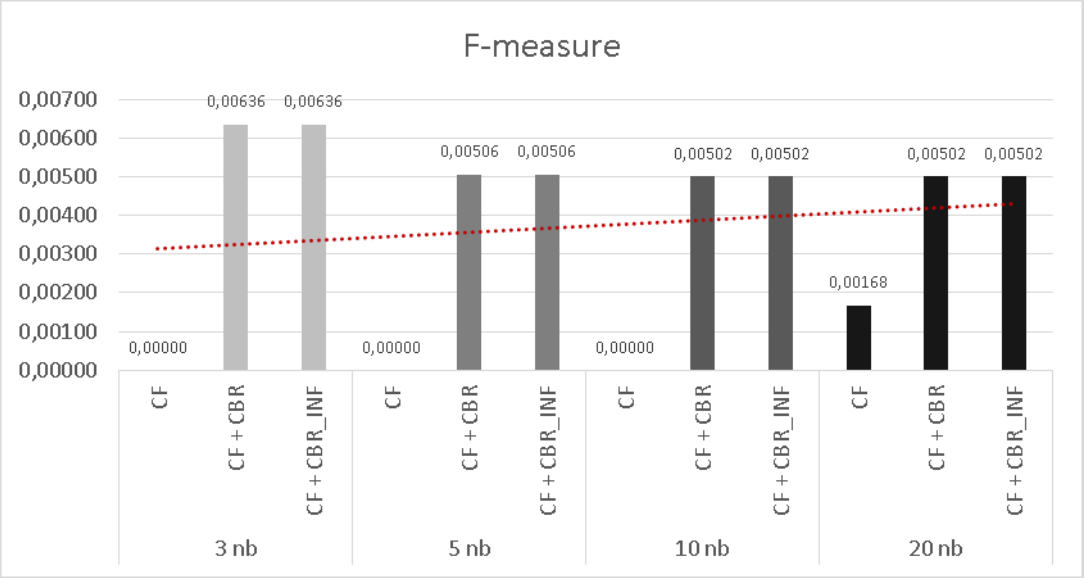
Şekil 7.37, Şekil 7.38, Şekil 7.39 tüm kullanıcı-şarkı verisinin %10' u kullanılarak elde edilen 10 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.37 Kullanıcı - Şarkı verisi (%10 eğitim - 10 öneri) precision grafiği



Şekil 7.38 Kullanıcı - Şarkı verisi (%10 eğitim - 10 öneri) recall grafiği

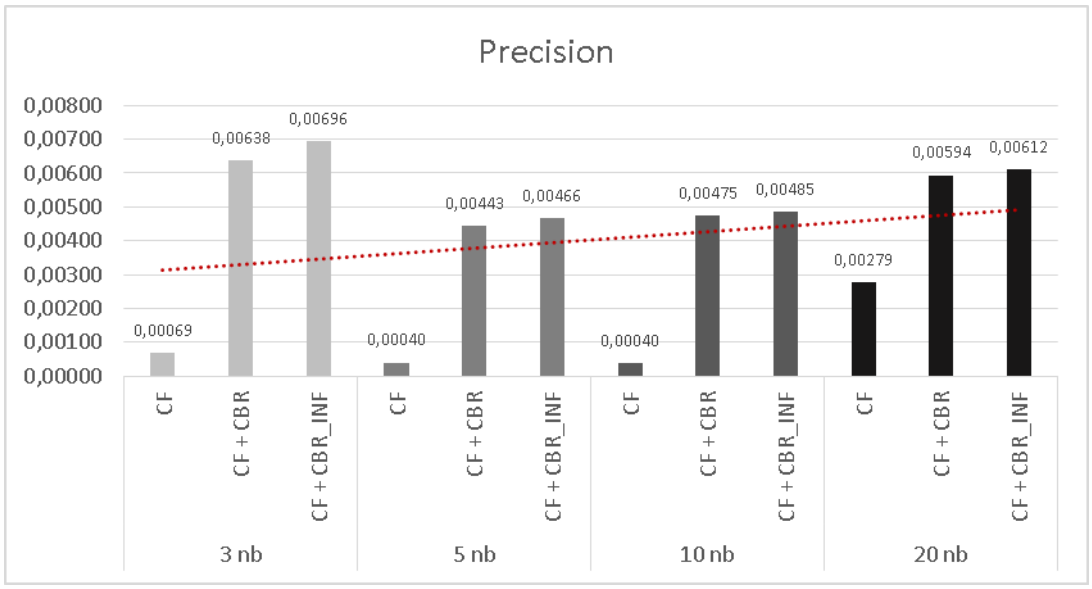


Şekil 7.39 Kullanıcı - Şarkı verisi (%10 eğitim - 10 öneri) f-measure grafiği

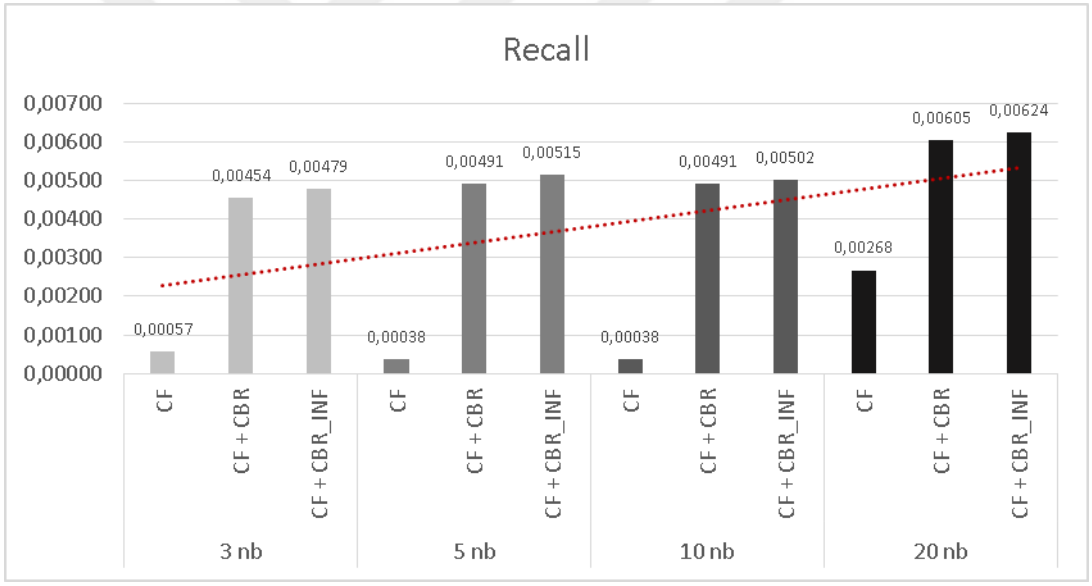
Kullanıcı - Şarkı verisinin % 10'u kullanılarak 10 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda artış gösterdiği izlenmektedir. Ancak ontolojik çıkarımlı durum tabanlı çıkarsama metodu sonuçları değiştirememektedir. Yapılan testler 3, 5, 10 ve 20 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir. Elde edilen değerlerin düşük olmasının sebebi veri çeşitliliğin çok fazla olmasından kaynaklanmaktadır.

7.1.3.2 10 Öneri Sonuçları (%50 Eğitim Verisi)

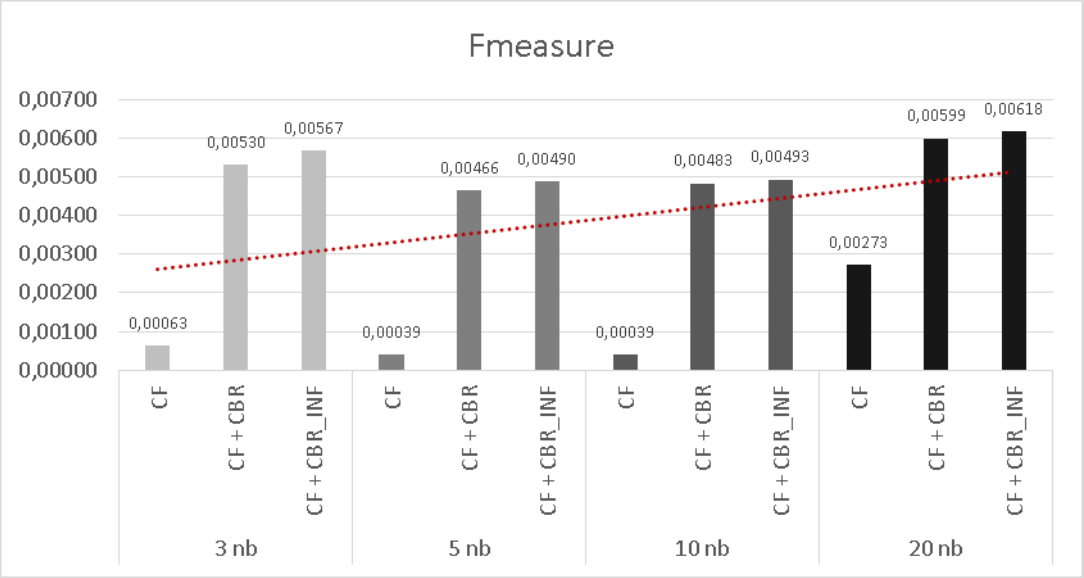
Şekil 7.40, Şekil 7.41, Şekil 7.42 tüm kullanıcı-şarkı verisinin %50' si kullanılarak elde edilen 10 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.40 Kullanıcı - Şarkı verisi (%50 eğitim - 10 öneri) precision grafiği



Şekil 7.41 Kullanıcı - Şarkı verisi (%50 eğitim - 10 öneri) recall grafiği

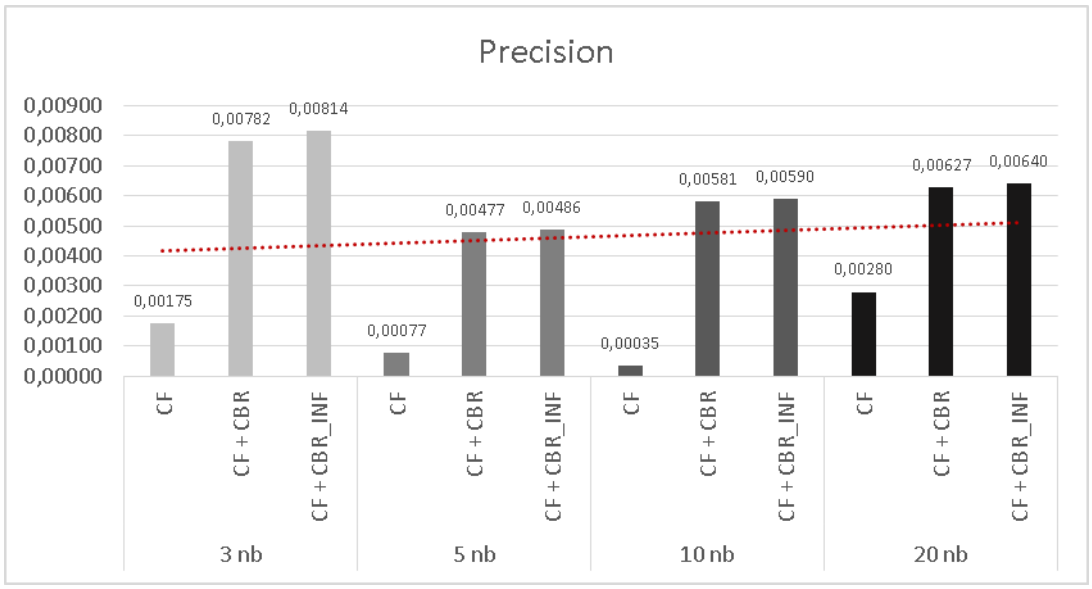


Şekil 7.42 Kullanıcı - Şarkı verisi (%50 eğitim - 10 öneri) f-measure grafiği

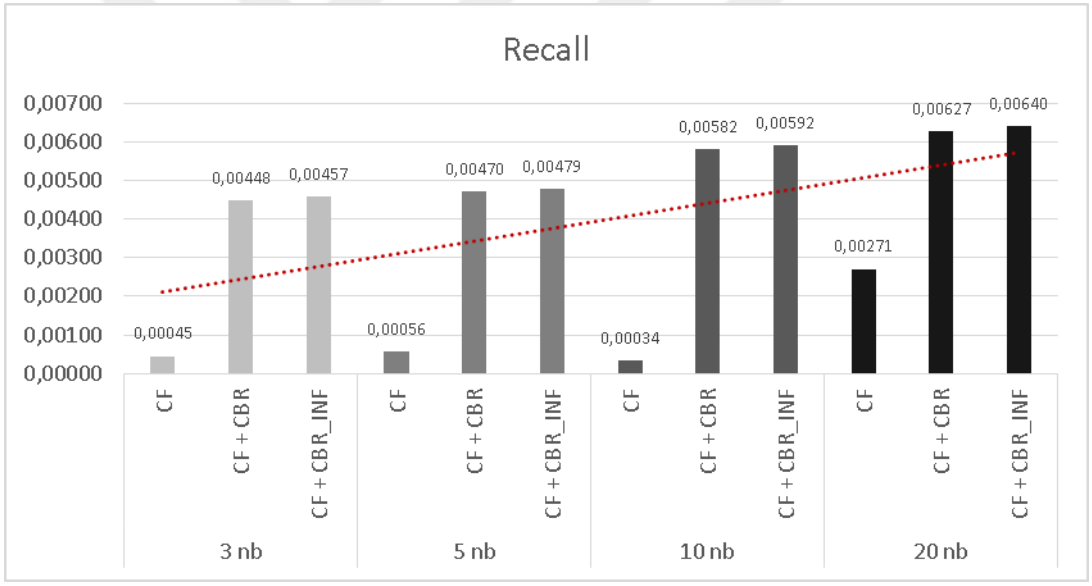
Kullanıcı - Şarkı verisinin % 50'si kullanılarak 10 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda artış gösterdiği izlenmektedir. Yapılan testler 3, 5, 10 ve 20 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir. Elde edilen değerlerin düşük olmasının sebebi veri çeşitliliğin çok fazla olmasından kaynaklanmaktadır.

7.1.3.3 10 Öneri Sonuçları (%90 Eğitim Verisi)

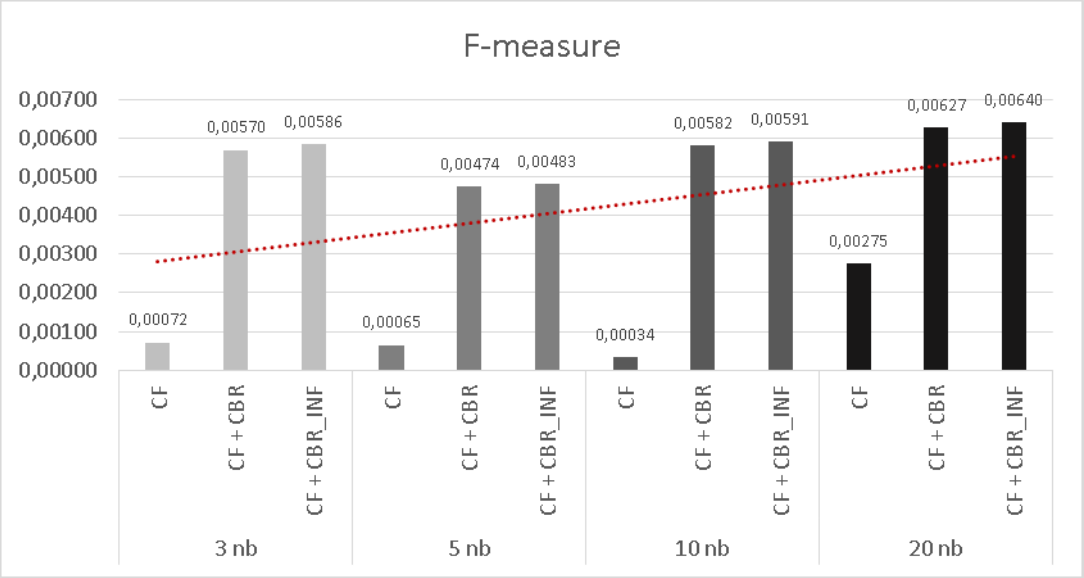
Şekil 7.43, Şekil 7.44, Şekil 7.45 tüm kullanıcı-şarkı verisinin %90' ı kullanılarak elde edilen 10 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.43 Kullanıcı - Şarkı verisi (%90 eğitim - 10 öneri) precision grafiği



Şekil 7.44 Kullanıcı - Şarkı verisi (%90 eğitim - 10 öneri) recall grafiği

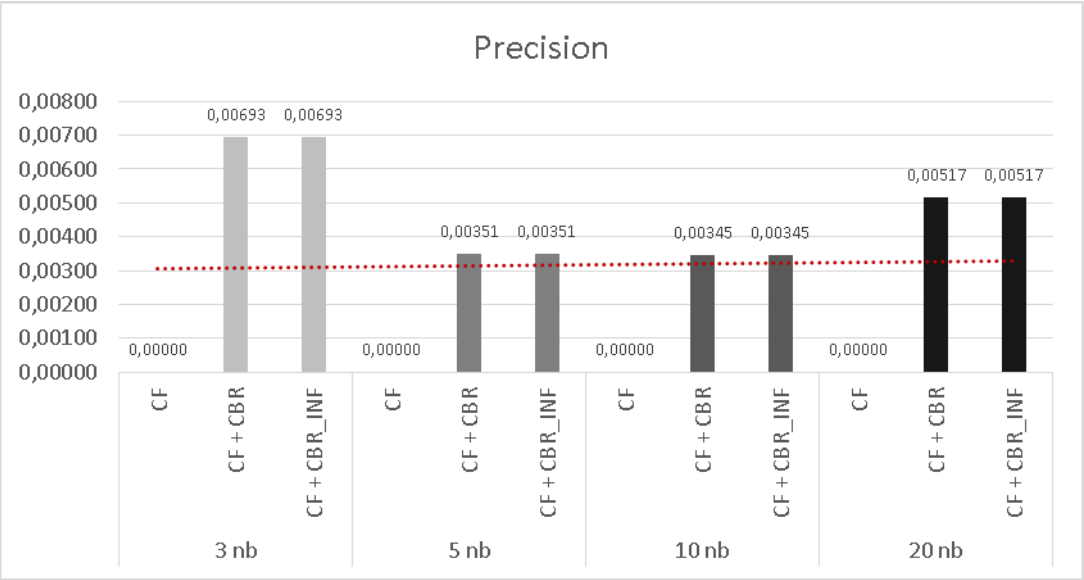


Şekil 7.45 Kullanıcı - Şarkı verisi (%90 eğitim - 10 öneri) f-measure grafiği

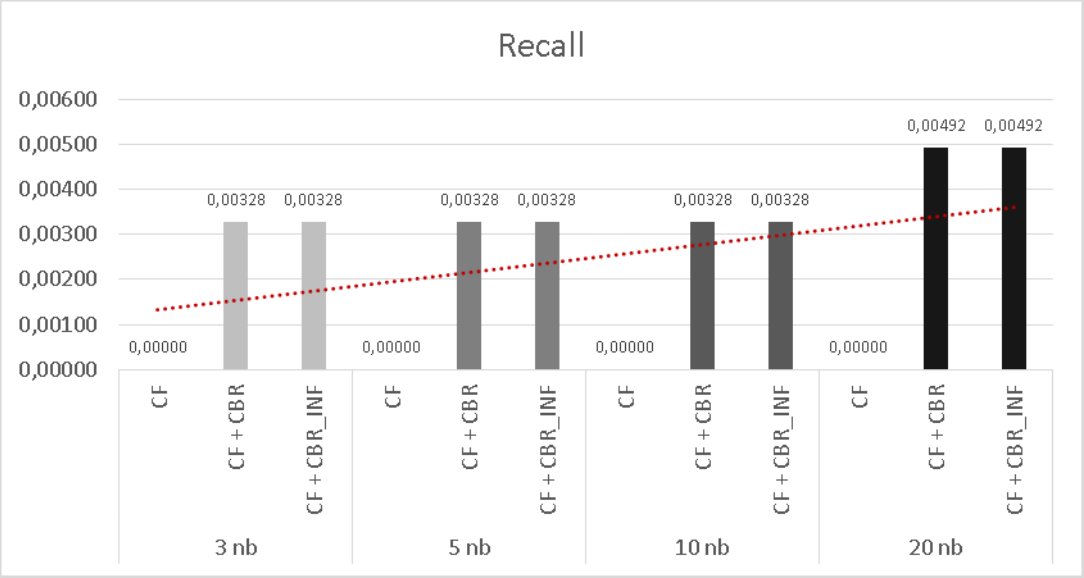
Kullanıcı - Şarkı verisinin % 90'ı kullanılarak 10 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda artış gösterdiği izlenmektedir. Yapılan testler 3, 5, 10 ve 20 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir. Elde edilen değerlerin düşük olmasının sebebi veri çeşitliliğin çok fazla olmasından kaynaklanmaktadır.

7.1.3.4 20 Öneri Sonuçları (%10 Eğitim Verisi)

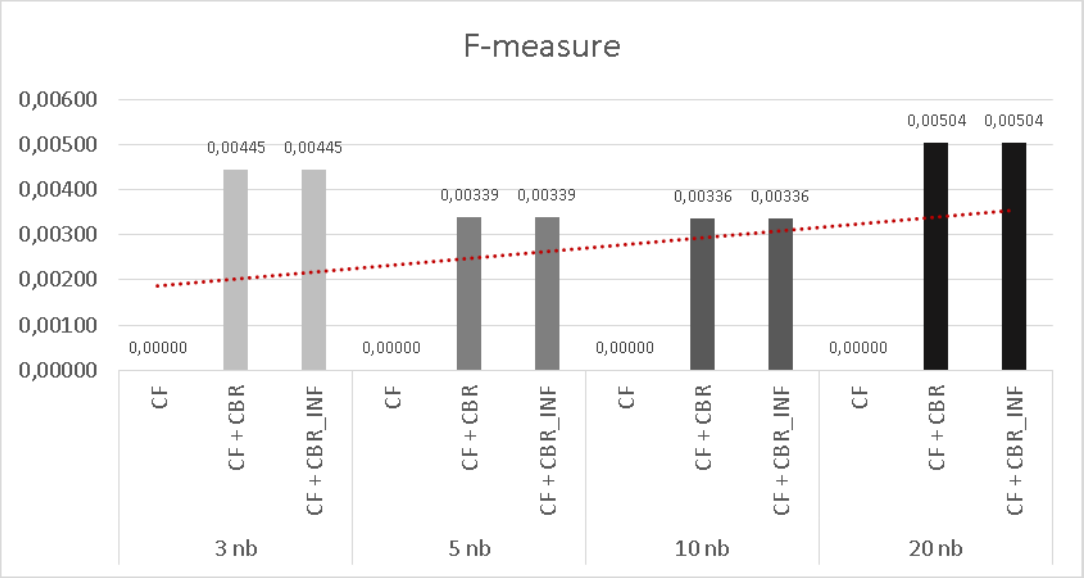
Şekil 7.46, Şekil 7.47, Şekil 7.48 tüm kullanıcı-şarkı verisinin %10' u kullanılarak elde edilen 20 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.46 Kullanıcı - Şarkı verisi (%10 eğitim - 20 öneri) precision grafiği



Şekil 7.47 Kullanıcı - Şarkı verisi (%10 eğitim - 20 öneri) recall grafiği

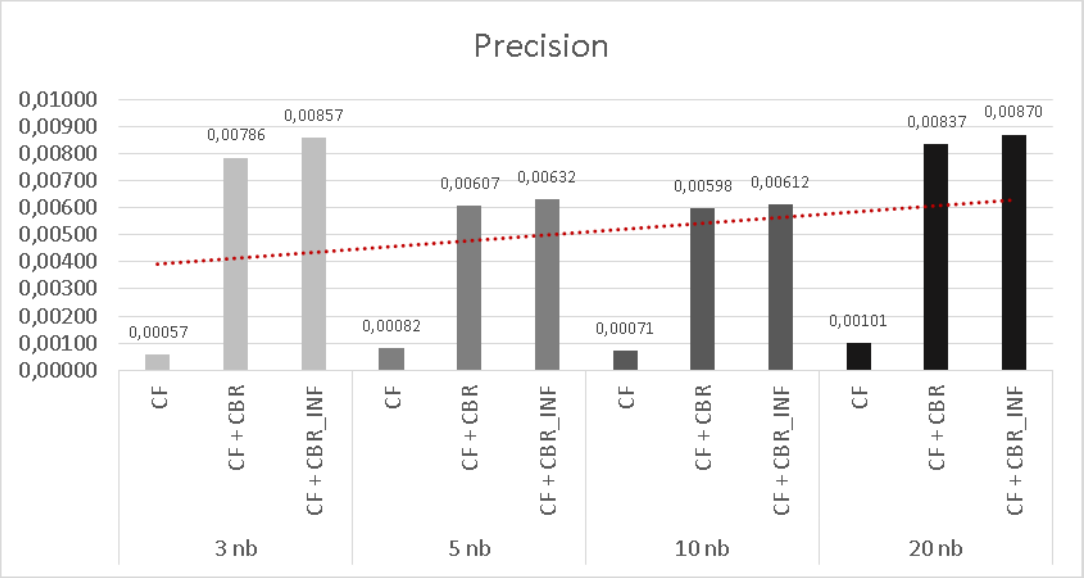


Şekil 7.48 Kullanıcı - Şarkı verisi (%10 eğitim - 20 öneri) f-measure grafiği

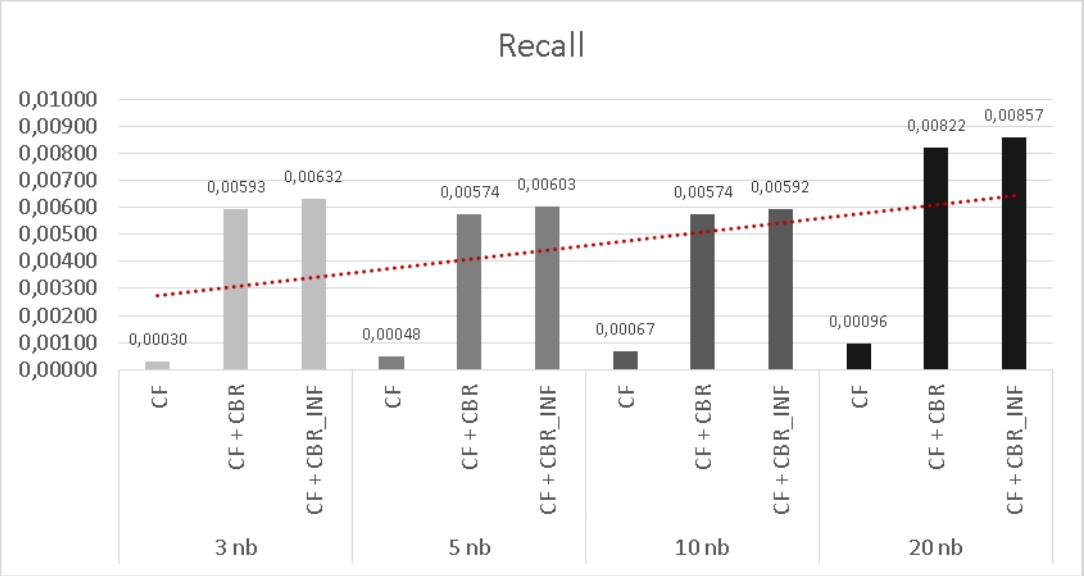
Kullanıcı - Şarkı verisinin % 10'u kullanılarak 20 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda artış gösterdiği izlenmektedir. Ancak ontolojik çıkarımlı durum tabanlı çıkarsama metodu sonuçları değiştirememektedir. Yapılan testler 3, 5, 10 ve 20 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir. Elde edilen değerlerin düşük olmasının sebebi veri çeşitliliğin çok fazla olmasından kaynaklanmaktadır.

7.1.3.5 20 Öneri Sonuçları (%50 Eğitim Verisi)

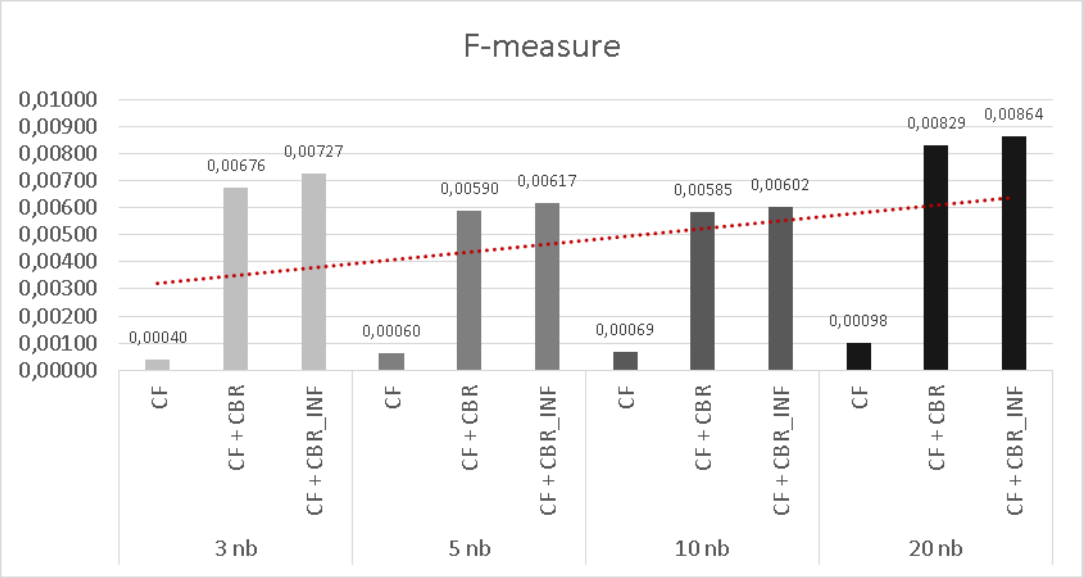
Şekil 7.49, Şekil 7.50, Şekil 7.51 tüm kullanıcı-şarkı verisinin %50' si kullanılarak elde edilen 20 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.49 Kullanıcı - Şarkı verisi (%50 eğitim - 20 öneri) precision grafiği



Şekil 7.50 Kullanıcı - Şarkı verisi (%50 eğitim - 20 öneri) recall grafiği

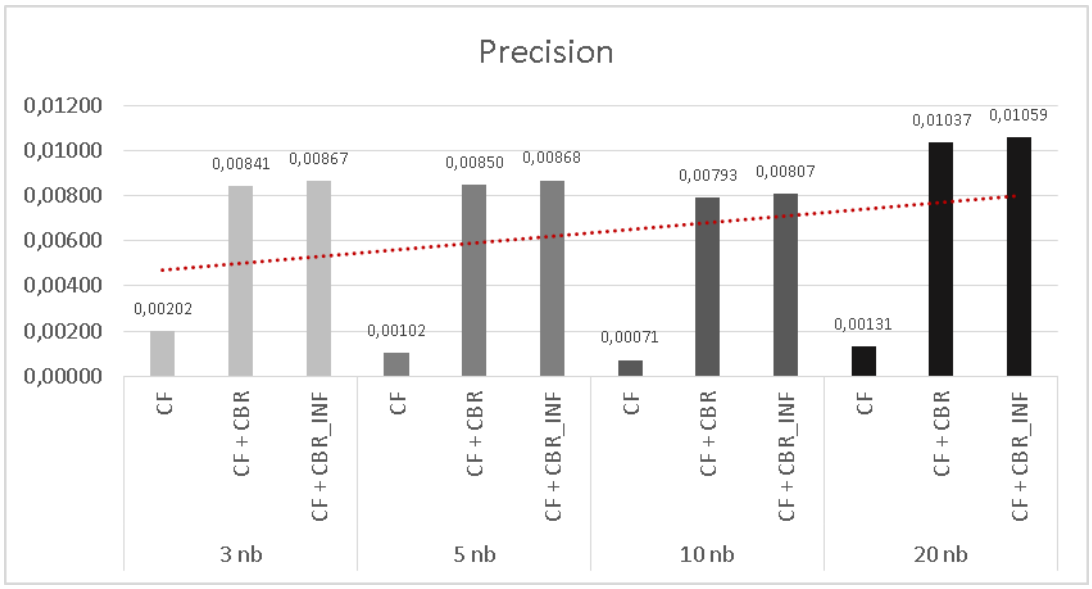


Şekil 7.51 Kullanıcı - Şarkı verisi (%50 eğitim - 20 öneri) f-measure grafiği

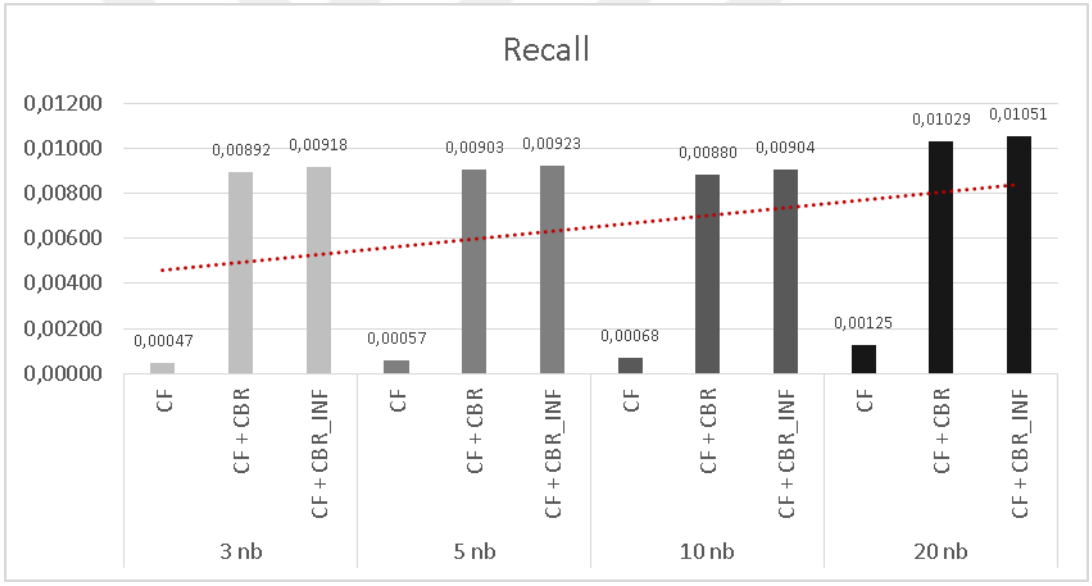
Kullanıcı - Şarkı verisinin % 50'si kullanılarak 20 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda artış gösterdiği izlenmektedir. Yapılan testler 3, 5, 10 ve 20 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir. Elde edilen değerlerin düşük olmasının sebebi veri çeşitliliğin çok fazla olmasından kaynaklanmaktadır.

7.1.3.6 20 Öneri Sonuçları (%90 Eğitim Verisi)

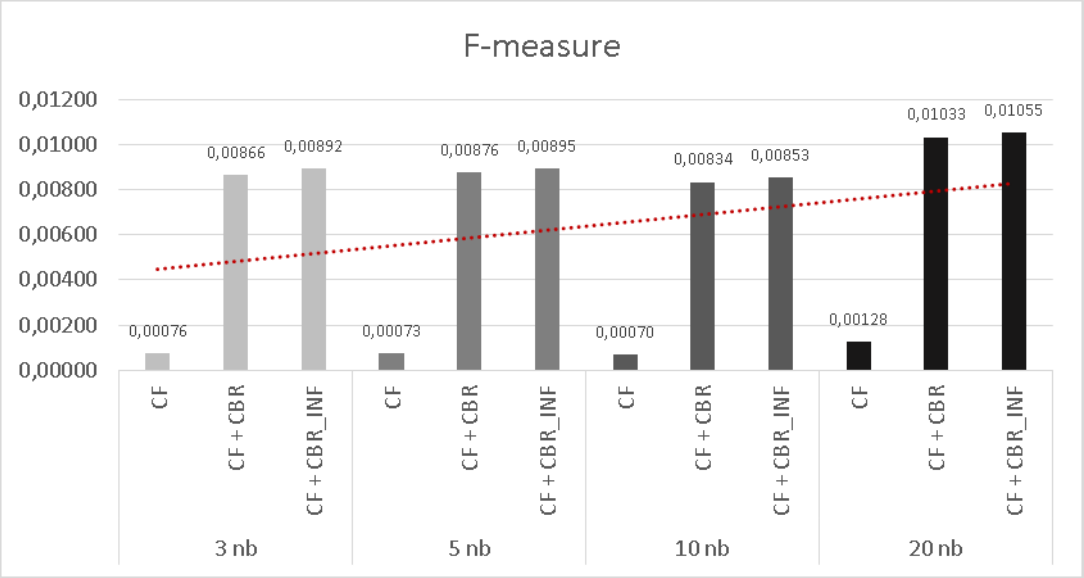
Şekil 7.52, Şekil 7.53, Şekil 7.54 tüm kullanıcı-şarkı verisinin %90' ı kullanılarak elde edilen 20 önerimin farklı komşuluklardaki precision, recall ve f-measure grafiklerini göstermektedir.



Şekil 7.52 Kullanıcı - Şarkı verisi (%90 eğitim - 20 öneri) precision grafiği



Şekil 7.53 Kullanıcı - Şarkı verisi (%90 eğitim - 20 öneri) recall grafiği



Şekil 7.54 Kullanıcı - Şarkı verisi (%90 eğitim - 20 öneri) f-measure grafiği

Kullanıcı - Şarkı verisinin % 90'ı kullanılarak 20 önerim yapılarak elde edilen sonuçlardan precision, recall ve f-measure değerlerinin önerilen yöntembilim kullanıldığı durumlarda artış gösterdiği izlenmektedir. Yapılan testler 3, 5, 10 ve 20 komşuluk için tekrarlanmış ve artışın durumun değişmediği gözlemlenmektedir. Elde edilen değerlerin düşük olmasının sebebi veri çeşitliliğin çok fazla olmasından kaynaklanmaktadır.

7.2 Değerlendirme

Bu bölümde elde ettiğimiz sonuçlar doğrultusunda, araştırmanın başında belirlediğimiz sorulara verdiğimiz yanıtları sunuyoruz.

7.2.1 Araştırma Sorularına Verilen Yanıtlar

Araştırma sorularına verilen yanıtlar aşağıda listelenmektedir.

7.2.1.1 Tavsiye sistemlerinde kullanılan işbirlikçi filtreleme yöntemi kullanılırken kullanıcı-ürün önerileri yapılırken karşılaşılan soğuk başlangıç problemi nasıl çözülebilir?

Elde ettiğimiz sonuçlar klasik işbirlikçi filtreleme yönteminin Durum Tabanlı Çıkarsama yöntemi ile birlikte kullanıldığında, soğuk başlangıç problemi oluştuğunda bile başarılı öneriler yapabildiğini göstermektedir. Bölüm 4' de sunduğumuz metodolojinin klasik

işbirlikçi filtreleme yöntemine göre daha başarılı sonuçlar ürettiğini, Bölüm 7.1' de sunduğumu değerlendirme sonuçları ile kanıtlamaktayız.

7.2.1.2 Ontoloji destekli Durumsal Çıkarsama Yöntemi ile farklılaşabilen kişisel profillere dayalı olarak öneriler yapabilen bir tavsiye sistemi nasıl geliştirilebilir?

Bu araştırma kapsamında, kişisel profile dayalı öneriler yapabilmek için bir profil ontolojisi geliştirilmiştir. Bu ontoloji bir müzik dinleyicisine ait kişisel özellikleri, isteklerini, demografik bilgileri, önceliklerini modellemektedir. Bu ontoloji sayesinde bir müzik dinleyicisinin özelliklerine ve tercihlerine en uygun müzik parçalarının önerilmesi sağlanabilmiştir. Bu açıdan, ortaya koyduğumuz ontoloji ve bu ontolojiyi kullanarak öneriler üreten Bölüm 4' de sunduğumuz hibrit tavsiye sistemi, kişiye özgün öneriler yapabilmektedir.

7.2.1.3 İşbirlikçi filtreleme ve Ontoloji Destekli Durumsal Çıkarsama Yönteminin birlikte kullanılması sonucu hibrit ve jenerik bir tavsiye sistemi için yazılım mimarisi nedir ve bu mimari yapı nasıl geliştirilmelidir?

Geliştirdiğimiz hibrit tavsiye sisteminin mimari yapısını Bölüm 4' de veriyoruz. Bu yöntemi içeren yazılımın, yazılım mimarisini Bölüm 5' de ve yazılımı gerçeklediğimiz prototip uygulamanın detaylarını Bölüm 6' da veriyoruz.

7.2.1.4 Muzik alanında kişilerin ilgilerini modelleyen bir ontoloji nasıl geliştirilir, nasıl olmalıdır?

Geliştirdiğimiz Ontolojiyi nasıl geliştirdiğimiz ve ontolojinin tüm bilgi varlıklarını, Bölüm 4' ün CBR verileri için, veri tanımlama modeli alt başlığı altında detaylı olarak anlatıyoruz.

7.2.1.5 Klasik İşbirlikçi Filtrelemeye Dayalı Tavsiye Yöntemi, Durumsal Çıkarsama Yöntemine dayalı Tavsiye Sistemi ve bu araştırmada önerilen Hibrit Tavsiye sistemi yöntemlerinin doğrulukları nasıl karşılaştırılmalıdır ve sonuçlar nasıl değerlendirilmelidir?

Bölüm 7.1' de, önerdiğimiz Hibrit Tavsiye Sisteminin başarısını nasıl ölçtüğümüzü ne gibi testler yaptığımızı detaylı olarak vermekteyiz.

BÖLÜM 8

SONUÇ ve ÖNERİLER

Bu çalışma kapsamında geliştirdiğimiz yöntem, ontoloji tabanlı CBR yöntemi ile CF yönteminin birlikte kullanımını içermektedir. Bu tezde, önerdiğimiz hibrit tavsiye sisteminin kullanılabilirliğini göstermek adına, müzik öneri veri kümesi üzerinde gerçekleştirilen test sonuçlarını ve yöntemi gerçekleyen müzik dinleme asistanı geliştirmede izlenen yöntemi raporlamaktayız. Yapılan testlerin sonuçları, önerilen ontoloji tabanlı CBR destekli CF tavsiye yönteminin, soğuk-başlangıç problemini ve kişisel öneri yapamama gibi zorlukları çözerek daha başarılı sonuçlar elde edilebileceğini işaret etmektedir. Ayrıca yapılan ontolojik veri zenginleştirme işlemleri ile birlikte yapılan önerimlerin başarımlarının arttırılabildiği gösterilmektedir.

Geliştirilen hibrit yöntem ile işbirlikçi tavsiye sistemlerinde karşılaşılan kullanıcı tercihlerine karşı oluşan duyarsızlık problemine de yeni bir çözüm önerilmektedir. Kullanıcıların müzik seçimini etkileyebilecek tercihler ve demografik bilgileri çıkarılarak bir ontolojik model üretilmiştir. Üretilen bu model üzerinden kullanıcı tercihlerine göre önerimler yapılmaktadır ve değişen kullanıcı tercihleri durum tabanlı çıkarsama motoru tarafından izlenebilmektedir.

Gelecekteki çalışmalar arasında, bu yöntemi kullanarak, farklı uygulama alanlarında da önerdiğimiz tavsiye yönteminin başarılı olduğunu göstermek yer almaktadır. E-ticaret, film, kitap vb. alanlarda önerilen hibrit yöntem ile ilgili alanda oluşturulucak bir ontoloji modeli kullanılarak farklı uygulamalar elde edilebilir.

KAYNAKLAR

[1] Dlab, M. ve Mezak, J., (2014). "Personalizing E-Learning 2.0 Using Recommendations", Methodologies and Intelligent Systems for Technology Enhanced Learning, 292:27-35.

[2] Vekariya, V. ve Kulkarni, G. R., (2012). "Hybrid Recommender Systems: Content-Boosted Collaborative Filtering for Improved Recommendations" 2012 International Conference on Communication Systems and Network Technologies, 36(4): 649–653.

[3] Sahebi, S. ve Cohen, W., (2011). "Community-Based Recommendations: a Solution to the Cold Start Problem", Recommender Systems and the Social Web, 23 October 2011, Chicago.

[4] Şahinkaya, F., (2010). A Content Boosted Collaborative Filtering Approach for Recommender Systems Based On Multi Level and Bidirectional Trust Data, Yüksek Lisans Tezi, ODTÜ Fen Bilimleri Enstitüsü, Ankara.

[5] YaLi, L. ve Pan, S. ve Xi, S., (2012). "A kind of Web Service Recommendation Method Based on Improved Hybrid Collaborative Filtering", Cognitive Informatics & Cognitive Computing, IEEE International Conference, 22-24 August 2012, Kyoto.

[6] Hameed, A. ve Jadaan, O., (2012). "Collaborative Filtering Based Recommendation System: A survey", International Journal on Computer Science and Engineering (IJCSE), 4: 859-876.

[7] Su, X. ve Khoshgoftaar, T. M., (2009). "A Survey of Collaborative Filtering Techniques", Advances in Artificial Intelligence, 2009: 1-19.

[8] Papagelis, M. ve Kutsuras, T., (2005). "Alleviating the Sparsity Problem of Collaborative Filtering Using Trust Inferences", iTrust'05 Proceedings of the Third International Conference on Trust Management, 23-26 May 2005, Berlin, 224-239.

[9] Burke, R., (2002). "Hybrid Recommender Systems: Survey and Experiments", User Modeling and User-Adapted Interaction, 12(4): 331–370.

[10] Hofmann, T., (2004). "Latent Semantic Models for Collaborative Filtering", ACM Transactions on Information Systems, 22(1): 89-115.

[11] Borchers, A., Herlocker, J., Konstan, J., ve Riedl, J.T., (1998). “Ganging up on Information Overload”, IEEE Computer, 31(4): 108.

[12] Yakut, I., (2008). Privacy Preserving Dimensionality Reduction-Based Collaborative Filtering, Yüksek Lisans Tezi, Anadolu Üniversitesi, Eskişehir.

[13] Kolodner, J.L., (1993). Case-Based Reasoning, Morgan Kaufmann Publishers, San Mateo.

[14] Shi, X. ve Yeh. A., (2001). “Case-Based Reasoning (CBR) in Development Control”, International Journal of Applied Earth Observation and Geoinformation, 3(3): 238-251.

[15] Gupta, U.G., (1994). “How Case-Based Reasoning Solves New Problems,” Interfaces, 24(6): 110-119.

[16] Schank, R.D., (1982). Dynamic Memory; a Theory of Reminding and Learning in Computers and People, Cambridge University Press, New York.

[17] Hammond, K., (1988). “Case-based Planning”, Proceedings of Workshop on Case-Based Reasoning, 4: 17-20.

[18] Aamodt, A. ve Plaza, E., (1994). “Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches”, AI Communications, 7(1): 39-59.

[19] Riesbeck, C.K. ve Schank, R.C., (1989). Inside Case-Based Reasoning, Lawrence Erlbaum Associates, Hillsdale.

[20] Yau, N.J. ve Yang, J.B., (1998). “Case Based Reasoning in Construction Management”, Computer-Aided Civil and Infrastructure Engineering. 13: 143-150.

[21] Karim, J., (2014). “Hybrid System for Personalized Recommendations”, Research Challenges in Information Science, 28-30 May 2014, 1-6.

[22] Linden, G., Smith, B. ve York, J., (2003). “Recommendations Item-to-Item Collaborative Filtering”, IEEE Internet Computing, 7(1): 76-80.

[23] Burke, R., (1999). “Integrating Knowledge-Based and Collaborative-Filtering Recommender Systems”, AAAI Technical Report, 1: 69-72.

[24] Meteren, R. ve Someren, M,. (2000). “Using Content Based Filtering for Recommendation”, Machine Learning in the New Information Age, 118(2): 10-14.

[25] Lops, P., Gemmis, M. ve Semeraro, G., (2011). “Content-Based Recommender Systems”, Recommender Systems Handbook, 7-105.

[26] Balanovic, M. ve Shoham, Y., (1997), “Content-Based, Collaborative Recommendation”, Communications of the ACM, 40: 88-89.

[27] Smyth, B., Hayes, C. ve Cunningham, P., (2001). “A Case-Based Reasoning View of Automated Colloborative Filtering”, 4th International Conference on Case-Based Reasoning, 10: 243-248.

[28] Chedrawy, Z., ve Abidi, S.S.R., (2006). “An Adaptive Personalization Recommendation Strategy Featuring Context-Sensitive Content Adaptation”, Adaptive Hypermedia and Adaptive Web-based Systems, 21-23 June 2006, 4018: 61-70.

[29] Araştırma için Kullanılan Lastfm Ham Müzik Veri Seti, http://www.dtic.upf.edu/~ocelma/MusicRecommendationDataset/lastfm-1K.html, 29 Mayıs 2016.

[30] Carter, PL., (2003). “Black Cultural Capital, Status Positioning and Schooling Conflicts For Low Income African American Youth”, Social Problems, 50(1): 136-155.

[31] White, G., (2001). The Effects of Class, Age, Gender and Race on Musical Preferences: An Examination of the Omnivore/Univore Framework, Master Of Science In Sociology, Faculty of the Virginia Polytechnic and State University, Virginia.

[32] LeBlanc, A., (1987). “The Development of Music Preference in Children”, Music and Child Development Springer New York, 137-157.

[33] Fox, W. ve Williams D.J., (2001). “Political Orientation and Music Preferences Among College Students”, Public Opinion Quarterly, 38(3): 352-371.

[34] Mizell, L., (2005). “Music Preferences in the U.S.: 1982-2002”, National Endowment for the Arts, 1-109.

[35] Apache Mahout, İşbirlikçi Filtreleme Yazılım Çatısı, http://mahout.apache.org/, 29 Mayıs 2016.

[36] Apache Jena, Ontoloji Yazılım Çatısı, http://jena.apache.org/, 29 Mayıs 2016.

[37] JColibri, Durum Tabanlı Çıkarım Yazılım Çatısı, http://gaia.fdi.ucm.es/research/colibri/jcolibri, 29 Mayıs 2016.

EK-A

MÜZİK ONTOLOJİSİ

Tez için geliştirilen müzik ontolojisi Çizelge A.1'de gösterilmektedir.

Çizelge A.1 Müzik ontolojisi için geliştirilen owl dosyası

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
   <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
   <!ENTITY dc "http://purl.org/dc/elements/1.1/" >
   <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
   <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
   <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
   <!ENTITY xsp "http://www.owl-ontologies.com/2005/08/07/xsp.owl#" >
]>


<rdf:RDF xmlns="http://www.example.com/ontologies/recommend.owl#"
    xml:base="http://www.example.com/ontologies/recommend.owl"
    xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:dc="http://purl.org/dc/elements/1.1/">
   <owl:Ontology rdf:about="http://www.example.com/ontologies/recommend.owl"/>


   <owl:ObjectProperty rdf:about="http://www.example.com/ontologies/recommend.owl#HAS-AGE">
      <rdfs:range rdf:resource="http://www.example.com/ontologies/recommend.owl#AGE"/>
      <rdfs:subPropertyOf rdf:resource="http://www.example.com/ontologies/recommend.owl#HAS-ATTRIBUTE"/>
      <rdfs:domain rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE"/>
   </owl:ObjectProperty>
```

Çizelge A.1 Müzik ontolojisi için geliştirilen owl dosyası (devamı)

```
<owl:ObjectProperty rdf:about="http://www.example.com/ontologies/recommend.owl#HAS-ARTIST">
  <rdfs:range rdf:resource="http://www.example.com/ontologies/recommend.owl#ARTIST"/>
  <rdfs:subPropertyOf rdf:resource="http://www.example.com/ontologies/recommend.owl#HAS-ATTRIBUTE"/>
  <rdfs:domain rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="http://www.example.com/ontologies/recommend.owl#HAS-ATTRIBUTE"/>

<owl:ObjectProperty rdf:about="http://www.example.com/ontologies/recommend.owl#HAS-COUNTRY">
  <rdfs:range rdf:resource="http://www.example.com/ontologies/recommend.owl#COUNTRY"/>
  <rdfs:subPropertyOf rdf:resource="http://www.example.com/ontologies/recommend.owl#HAS-ATTRIBUTE"/>
  <rdfs:domain rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="http://www.example.com/ontologies/recommend.owl#HAS-DURATION">
  <rdfs:range rdf:resource="http://www.example.com/ontologies/recommend.owl#DURATION"/>
  <rdfs:subPropertyOf rdf:resource="http://www.example.com/ontologies/recommend.owl#HAS-ATTRIBUTE"/>
  <rdfs:domain rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="http://www.example.com/ontologies/recommend.owl#HAS-GENDER">
  <rdfs:range rdf:resource="http://www.example.com/ontologies/recommend.owl#GENDER"/>
  <rdfs:subPropertyOf rdf:resource="http://www.example.com/ontologies/recommend.owl#HAS-ATTRIBUTE"/>
  <rdfs:domain rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="http://www.example.com/ontologies/recommend.owl#HAS-LISTENER">
  <rdfs:subPropertyOf rdf:resource="http://www.example.com/ontologies/recommend.owl#HAS-ATTRIBUTE"/>
  <rdfs:range rdf:resource="http://www.example.com/ontologies/recommend.owl#LISTENER"/>
  <rdfs:domain rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="http://www.example.com/ontologies/recommend.owl#HAS-LISTEN_WAY">
  <rdfs:subPropertyOf rdf:resource="http://www.example.com/ontologies/recommend.owl#HAS-ATTRIBUTE"/>
  <rdfs:range rdf:resource="http://www.example.com/ontologies/recommend.owl#LISTEN_WAY"/>
  <rdfs:domain rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="http://www.example.com/ontologies/recommend.owl#HAS-PLAY_COUNT">
  <rdfs:subPropertyOf rdf:resource="http://www.example.com/ontologies/recommend.owl#HAS-ATTRIBUTE"/>
  <rdfs:range rdf:resource="http://www.example.com/ontologies/recommend.owl#PLAY_COUNT"/>
```

Çizelge A.1 Müzik ontolojisi için geliştirilen owl dosyası (devamı)

```
    <rdfs:domain rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="http://www.example.com/ontologies/recommend.owl#HAS-REGISTER">
  <rdfs:subPropertyOf rdf:resource="http://www.example.com/ontologies/recommend.owl#HAS-ATTRIBUTE"/>
  <rdfs:domain rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE"/>
  <rdfs:range rdf:resource="http://www.example.com/ontologies/recommend.owl#REGISTER"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="http://www.example.com/ontologies/recommend.owl#HAS-RELEASE_DATE">
  <rdfs:subPropertyOf rdf:resource="http://www.example.com/ontologies/recommend.owl#HAS-ATTRIBUTE"/>
  <rdfs:domain rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE"/>
  <rdfs:range rdf:resource="http://www.example.com/ontologies/recommend.owl#RELEASE_DATE"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="http://www.example.com/ontologies/recommend.owl#HAS-SELF_VIEW">
  <rdfs:subPropertyOf rdf:resource="http://www.example.com/ontologies/recommend.owl#HAS-ATTRIBUTE"/>
  <rdfs:domain rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE"/>
  <rdfs:range rdf:resource="http://www.example.com/ontologies/recommend.owl#SELF_VIEW"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="http://www.example.com/ontologies/recommend.owl#HAS-TAG">
  <rdfs:subPropertyOf rdf:resource="http://www.example.com/ontologies/recommend.owl#HAS-ATTRIBUTE"/>
  <rdfs:domain rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE"/>
  <rdfs:range rdf:resource="http://www.example.com/ontologies/recommend.owl#TAG"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="http://www.example.com/ontologies/recommend.owl#HAS-TRACK">
  <rdfs:subPropertyOf rdf:resource="http://www.example.com/ontologies/recommend.owl#HAS-ATTRIBUTE"/>
  <rdfs:domain rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE"/>
  <rdfs:range rdf:resource="http://www.example.com/ontologies/recommend.owl#TRACK"/>
</owl:ObjectProperty>

<owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#ADAPTATION">
  <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#SOLUTION"/>
</owl:Class>

<owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#AGE">
  <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PROFILE"/>
</owl:Class>
```

Çizelge A.1 Müzik ontolojisi için geliştirilen owl dosyası (devamı)

```
<owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#ARTIST">
  <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PREFERENCE"/>
</owl:Class>

<owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#CBR_DESCRIPTION"/>

<owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#CBR_INDEX"/>

<owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#COUNTRY">
  <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PROFILE"/>
</owl:Class>

<owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#DURATION">
  <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PREFERENCE"/>
</owl:Class>

<owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#ENVIRONMENT">
  <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PROBLEM"/>
</owl:Class>

<owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#FEEDBACK">
  <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#RESULT"/>
</owl:Class>

<owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#GENDER">
  <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PROFILE"/>
</owl:Class>

<owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#LISTENER">
  <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PREFERENCE"/>
</owl:Class>

<owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#LISTEN_WAY">
  <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#ENVIRONMENT"/>
</owl:Class>

<owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#OTHER_SOLUTION">
  <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#RESULT"/>
</owl:Class>

<owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#PLAY_COUNT">
```

Çizelge A.1 Müzik ontolojisi için geliştirilen owl dosyası (devamı)

```
<rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PREFERENCE"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#PREFERENCE">
    <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PROBLEM"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#PROBLEM">
    <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#CBR_INDEX"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#PROFILE">
    <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PROBLEM"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#QUERY">
    <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PROBLEM"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#RECOMMENDATION">
    <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#SOLUTION"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#RECOMMEND_CASE">
    <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#CBR_DESCRIPTION"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#REGISTER">
    <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PROFILE"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#RELEASE_DATE">
    <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PREFERENCE"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#RESULT">
    <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#CBR_INDEX"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#SELF_VIEW">
    <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PREFERENCE"/>
```

Çizelge A.1 Müzik ontolojisi için geliştirilen owl dosyası (devamı)

```
</owl:Class>

  <owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#SOLUTION">
    <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#CBR_INDEX"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#TAG">
    <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#PREFERENCE"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.example.com/ontologies/recommend.owl#TRACK">
    <rdfs:subClassOf rdf:resource="http://www.example.com/ontologies/recommend.owl#RECOMMENDATION"/>
  </owl:Class>
</rdf:RDF>
```

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Zafer DÜZEN

Doğum Tarihi ve Yeri : 20/09/1988 - İstanbul

Yabancı Dili : İngilizce

E-posta : duzenz@gmail.com

ÖĞRENİM DURUMU

Derece	**Alan**	**Okul/Üniversite**	**Mezuniyet Yılı**
Y. Lisans	Bilgisayar Müh.	Yıldız Teknik Üniversitesi	2016
Lisans	Matematik Müh.	İTÜ	2011
Lise	Fen Bilimleri	Samiha Ayverdi Anadolu L.	2006

İŞ TECRÜBESİ

Yıl	**Firma/Kurum**	**Görevi**
2014- Devam Ediyor	Huawei	Yazılım Mühendisi
2011-2014	Kartaca	Yazılım Mühendisi