YBS Ansiklopedi

www.YBSAnsiklopedi.com

Cilt 3 Sayı 3, Eylül 2016

Duygu Analizi (Sentimental Analysis)

Sadi Evren SEKER

Smith College, Department of Computer Science, MA, US

Özet

Bu yazının amacı, literatürde duygu analizi olarak geçen kavramı açıklamaktır. Genel olarak bu alanda kullanılan terimler, özellik çıkarım yöntemleri, karşılaşılan ve çözülmesi güç görülen zorluklar ve büyük veri dünyasındaki çözüm önerileri gösterilmeye çalışılmıştır. Yazıda büyük veri dünyasında kullanılan Spark isimli yazılımın üzerinde çalışan MLLib isimli makine öğrenme kütüphanesinden örnek kodlar verilerek kullanılan algoritmalar açıklanmıştır.

Anahtar Kavramlar: *Duygu Analizi, Büyük Veri, Makine Öğrenmesi*

Abstract

Purpose of this paper is briefly introducing the concept of sentimental analysis. Terminology about the sentimental analysis, feature extraction, some difficulties in the sentimental analysis and some solutions in the big data, are introduced in the paper. In this paper, some code samples from MLLib library running on Spark is also given and their purpose and usage is explained.

Keywords: *Sentimental Analysis, Big Data, Machine Learning*

1. Sentimental Analysis (Duygu Analizi) Nedir?

Duygu analizi temel olarak bir metin işleme (text processing) işlemi olup verilen metinin duygusal olarak ifade etmek istediği sınıfı belirlemeyi amaçlar. Duygu analizinin ilk çalışmaları duygusal kutupsallık (sentimental polarity) olarak geçmekte olup verilen metni olumlu (positive), olumsuz (negative) ve nötr olarak sınıflandırmayı amaçlamaktadır.

Daha sonraları çalışmalar farklı duygu durumlarını belirten analizlere de izin vermiştir. Duygu durumlarını kodlamak için genelde sınıf etiketlemesi yaklaşımına benzer şekilde veri kümesi üzerinde her metnin tek bir duygu ile etiketlenmesi veya metinlerde birden fazla duygunun etiketlenmesi şeklinde yaklaşımlar görülmektedir.

Literatürde bütün bu çalışmalar enformasyon getirme (information retrieval) alanı altında çalışılmakta olup, daha sonraları çalışmaların yerini, duygu analizindeki metin-etiket bağlantısının dışına çıkarak açık uçlu sonuçlara dönüştüğü fikir madenciliği (opinion mining) çalışmalarına bıraktığı görülmüştür. Bu çalışma kapsamında duygu analizinin etraflıca anlaşılması ve yaşanan problemlerin ayrıştırılması için bilgi (knowledge) seviyesi problemlerden bahsedilecek olsa da sonuçta projenin kapsamı enformasyon (information) seviyesi problemlerin çözümü ile kısıtlıdır.

Sosyal ağlardaki problemlerden birisi de fikir veya kanaat madenciliği olarak literatürde geçen problemdir. Problem literatürdeki konumu itibariyle duygu analizi (sentimental analysis) altında geçmektedir. Buna göre bir sosyal medya bilgisinin (mesaj, paylaşım, duvar yazısı, haber v.s.) taşımış olduğu fikri anlambilimsel olarak göstermek için yapılan çalışmaya fikir madenciliği denir. Fikir madenciliği için en önemli kriterlerden birisi, fikir veya kanaat oluşumunun bir topluluk üzerinde inceleniyor olmasıdır. Özel olarak seçilmiş uzmanların fikirleri alınmadığı sürece fikir madenciliğinin ulaşmak istediği sonuç, bir topluluktaki bütün bireylerin fikirlerini anlayabilmektir. Ne yazık ki bütün bireyler ulaşmanın imkansızlığı yüzünden genelde bu işlem örneklemelerle yapılmaktadır. Örneğin anket çalışmaları bu örneklemelerden birisidir.

Fikir madenciliği sırasında unutulmaması gereken konulardan birisi de fikirlerin kişisel olduğudur. Yani doğru veya yanlış bir fikir aranmaz, fikir madenciliği mevcut durumun tespitine çalışır.

Genelde literatürde ilk uygulamaları ve halen üzerinde en çok çalışılan uygulama duygusal kutupsallıktır (sentimental polarity). Bu problemde metinleri duygusal olarak olumlu veya olumsuz şeklinde iki gruba ayrılmaya çalışılır. Örneğin bir siyasi parti, bir futbol takımı veya bir televizyon programı hakkında sosyal medyada yapılan yorumların tamamını bir bilgisayarın inceleyerek doğal dil işleme [1] ve metin madenciliği teknikleri [2]ile bu konularda yapılan yorumların olumlu veya olumsuz olarak sınıflandırılması mümkündür.

Fikir madenciliği için çok farklı yöntemler geliştirilmiştir. Örneğin kelimelerin olumlu veya olumsuz olarak ayrılması ve yorumlarda geçen kelimelerin sayılarına göre yorumların olumlu veya olumsuz olarak sınıflandırılması ilk ve en basit yöntemlerden birisidir. Ancak alaycı yorumlar düşünüldüğünde bu yöntemin başarı oranının göreceli olarak düşük olduğu görülecektir. Bu yüzden kendi kendini geliştiren ve kelimelerin anlamlarını kendisi bulan daha zeki yöntemler geliştirilmiştir.

Duygu analizi ve fikir madenciliğinin dayandığı ilk sınıflandırma ve metin madenciliği teknikleri aslında iki sınıflı basit problemler olarak görülebilir. Örneğin uzun yıllar çok benzer metin madenciliği teknikleri kullanılarak e-postaların istenmeyen veya zararlı e-postalar veya zararsız e-postalar olarak iki sınıfa ayrılması üzerinde çalışıldı [3].

Günümüzde problemler daha karmaşık haller almaktadır. Örneğin bir filmi izleyen iki farklı kişinin görüşleri arasındaki farkın ne kadarının filim oyuncularından kaynaklandığı, filmin yönetmeninin veya senaryo yazarının izleyicilerin görüşüne etkisi gibi daha karmaşık çıkarımlar artık sorgulanabilmektedir.

Veya bir internet satış mağazasında ürünler hakkında yapılan yorumların ne kadarı satıcının etkisi ile ne kadarı kargo şirketinden ne kadarı ürünün kendisinden kaynaklanmaktadır gibi sorular artık birer çalışma alanı olarak belirmektedir.

Fikir madenciliği çalışmalarında aşağıdaki sorulara cevap aranabilir:

- **Öznellik sınıflandırması (subjectivity classification):** Verilen bir metnin/cümlenin herhangi bir fikir içerip içermediğine bakılır.
- **Duygusal sınıflandırma:** Verilen bir cümlenin olumlu/olumsuz veya nötr olması durumunu bulmaya çalışır.
- **Fikrin yardımcı olma ihtimali:** Herhangi bir fikir içeren metnin kişilere ne kadar yardımcı olacağını bulmaya çalışır (daha çok alışveriş sitelerindeki yorumlar için kullanılmaktadır).
- **İstenmeyen fikir taraması:** Bir fikrin kötü amaçla yazılması durumunu tespit için kullanılır. Örneğin bir yorum yazısının reklam içermesi, okuyucuyu yönlendirici olması gibi durumları bulmaya çalışır.
- **Fikir özetleme:** Çok sayıdaki fikrin veya uzun bir fikir metninin kısa bir şekilde ifade edilmesi için çalışır.

Örneğin anahtar önemdeki cümlelerin çıkarılması, ürün veya bakış açısına göre sınıflandırılması gibi problemleri bulunmaktadır.

- **Karşılaştırmalı fikirlerin çıkarılması:** Örneğin iki veya daha fazla ürün veya kavramın karşılaştırıldığı fikirlerde, metnin hangi kavramı hangi açıdan karşılaştırdığı ve bu karşılaştırmaya göre kavramların görece pozisyonlarını belirlemekte kullanılırlar.

Fikir madenciliği çalışmaları üç farklı açıdan incelenebilir, bunlar:

- Bakış tabanlı (veya özellik tabanlı) fikir madenciliği teknikleri
- Frekans veya ilişki tabanlı fikir madenciliği teknikleri
- Model tabanlı fikir madenciliği teknikleri

Olarak sayılabilir. Yazının devamında her bir yaklaşım ayrı bir alt başlıkta incelenecektir.

1.1. Bakış Tabanlı / Özellik Tabanlı Fikir Madenciliği

Bu yaklaşımda fikir madenciliği ilgili metin üzerinden hangi bakış açısında fikir beyan edildiğini bulmaya çalışır. Örneğin bir fotoğraf makinesi ile ilgili yapılmış bir yorumda, yorumu yazan kişi amatör bir fotoğrafçı veya profesyonel bir fotoğrafçı olabilir, dolayısıyla kişinin yorumunun kimin işine daha çok yarayacağının anlaşılabilmesi için yorumu yazan kişinin hangi bakış açısıyla yorumu yazdığının anlaşılması önemlidir. Benzer şekilde fotoğraf makinesi örneğinden devam edilirse, fotoğraf makinesinin malzemesinin kalitesi, çektiği resimlerin kalitesi, renk ayarı, fotoğrafçının eline uyumu (tabi bu durumda ince veya şişman parmaklı bir fotoğrafçı oluşu), kasasının rengi, hafifliği, boyutları, üzerindeki yazılımın kullanım kolaylığı, diğer objektif üreticileri ile uyumluluğu gibi onlarca farklı açıdan incelenmesi mümkündür. Kısaca bir fotoğraf makinesi hakkında olumlu veya olumsuz yorum yapılması duygusal kutupsallık açısından önemli olmakla birlikte hangi bakış açısında göre olumlu veya olumsuz olduğunu inceleyen çalışmalara bakış tabanlı (aspect-based) fikir madenciliği ismi verilmektedir.

Bakış tabanlı fikir madenciliğindeki önemli konulardan birisi de karşılaştırmalı fikirlerin çıkarılmasıdır. Örneğin fotoğraf makinesi hakkında yapılan bir yorumun, aynı fotoğraf makinesinin bir önceki yıl çıkan versiyonuna göre veya rakip firmanın benzer ürününe göre veya bir üst sınıftaki fotoğraf makinesine göre veya cep telefonunun kamerasına göre olumlu veya olumsuz olmasının yanında hangi açılardan olumlu veya olumsuz olduğunun da incelenmesi gerekir. Örneğin bir fotoğraf makinesini güvenlik kamerasına göre çok daha kolay hareket ettirilebildiğinin söylenmesi ile cep telefonuna göre çok daha kolay hareket ettirilebildiğinin söylenmesi arasında ifade açısından fark vardır.

1.2. Frekans / İlişki Tabanlı Fikir Madenciliği

Metinlerin üzerinden amaca yönelik olarak fikir çıkarımı yapılırken kelime sayısı, isim, sıfat, zarf veya fiil gibi kelimelerin sıklıkları (frekansları) üzerinden fikir madenciliği yapılmasına verilen isimdir. Genelde fikirlerin bu kelimeler ile ifade edildiği kabulüne dayanmaktadır. Örneğin 2011 yılında yapılan bir araştırmada fikirlerin %60-70 gibi önemli bir oranının metindeki isimlere dayandığı bulunmuştur [4]. Yine istisnaları olmakla birlikte, bakış tabanlı fikir madenciliğinde de sık tekrar eden isimlerin kişilerin bakışını yansıtmakta olduğu bulunmuştur.

Frekans tabanlı fikir madenciliğinde, literatürde önemli yer tutan çalışmalardan birisi de özellik tabanlı özetleme çalışmalarıdır. Bu çalışmalarda öncelikle isim kelime grupları bulunarak bunlar uzunluklarına, kullanımdaki gerekliliklerine ve olumlu olumsuz kutupsallığına göre sınıflandırılmaktadır. Ardından bu isim kelime gruplarını tanımlayan sıfatlar kelime gruplarına eklenerek sık tekrar etmeyen duyguların elenmesi sağlanmaktadır [5].

Diğer bir yaklaşım ise belirli şablonların metin içerisinde aranarak belirli sonuçlara ulaşılmasıdır. Örneğin "harika X", "X özelliği ile gelmektedir", "X özelliği bulunmaktadır" gibi kalıplar aranarak X yerine gelen değerlerin birer bakış olarak aday gösterilmesi ve ardından kutupsallık analizleri ile bakış açısına göre fikir madenciliği yapılması mümkündür [6].

Frekans tabanlı fikir madenciliği ayrıca kelimelerin dilbilimsel özelliklerine göre etiketlenmeleri ile sağlanabilmektedir. Literatürde POS-Tagging olarak geçen kavram Türkçeye konuşmanın bir kısmının etiketlenmesi olarak çevrilebilir ve basitçe bir metindeki kelimelerin isim, sıfat ve hatta sayı sıfatı, özel isim gibi özelliklerine göre etiketlenmesi esasına dayanır. Bu etiketleme sürecinin fikir beyan edecek şekilde genişletilmesi de mümkündür. Örneğin "harika konumda" kelimeleri etiketlenirken "[güçlü][olumlu] konum" veya "yardımsever çalışanlar" kelimeleri etiketlenirken "[duygusal][olumlu]çalışan" şeklinde etiketlenmesi mümkündür. Daha sonra bu etiketlerin üzerinden yapılan sıklık analizleri ile fikir çıkarımı yapılması çok daha kolay hale gelecektir [7].

Literatürde ayrıca frekans ve ilişkiye dayalı fikir madenciliği çalışmalarının sağladığı karşılaştırmalı madencilik imkanları da bulunmaktadır. Örneğin bir sıfatın kuvvetinin belirli bir ölçeğe oturtulması mümkündür.

Mükemmel -> iyi -> ortalama -> zayıf -> kötü

Gibi bir sıralamada hepsi sıfat olmakla birlikte kuvvet dereceleri değişmekte ve dolayısıyla çıkarılan fikir açısından değişik oranlarda etki etmektedir [8].

Genel olarak frekans tabanlı yaklaşımların en büyük avantajı, uygulamadaki kolaylık ve çalışma sürelerindeki yüksek başarılarıdır. Bununla birlikte hata miktarları çok yüksek olup genelde elle müdahale edilerek ince ayarlarının yapılması gerekmektedir.

1.3. Model Tabanlı Fikir Madenciliği Teknikleri

Model tabanlı fikir madenciliği yöntemleri daha önceden etiketlenmiş ve dolayısıyla nasıl bir fikir içerdiği bilinen metinlerden oluşturduğu modelleri etiketlenmemiş metinlere uygulamakta ve bu sayede fikir çıkarımı yapılmamış metinlerden fikir çıkartmaya çalışmaktadır.

Aslında şimdiye kadar bahsi geçen diğer yöntemler de bu açıdan ele alındığında birer model tabanlı yaklaşım olarak kabul edilebilir ancak model tabanlı fikir madenciliğinin en belirgin özelliği gizli Markof Modelleri (HMM) veya koşullu rasgele alanlar (CRF) veya daha genel anlamda yapay sinir ağları (ANN) veya Bayez ağları (Bayesian Networks) gibi istatistiksel modellere dayanıyor olmasıdır. Modeller istatistiksel ağırlıklarını (çarpanlarını) etiketli metinlerde öğrenerek bu ağırlık değerlerine göre etiketsiz metinlerde fikir çıkarımı yapmaktadır.

Örneğin [9] çalışmalarında alt küme yaklaşımını kullanarak yeni bir gizli markof modeli geliştirmiş ve bakış ve duygu seviyesi değerlerin cümledeki konumuna göre ilişkileri üzerinden değişen ağırlıklara göre istatistiksel bir model oluşturmuştur. Benzer bir değerlendirme ise fikirlerin çıkarımının yapıldığı web sitelerinde bulunan fikirlere göre gruplamaya dayalı ve bu gruplar arasındaki ağırlıkları hesaplayan ve yine bir gizli markof modeli olan [10]'un çalışmasıdır.

Model tabanlı çalışmaların önemli bir kısmı ise başlıklara odaklanmaktadır. Bu çalışmaların yapmış olduğu kabul, metinlerde içerilen fikrin başlıkta daha net bir şekilde eyr alacağıdır. Bu yaklaşımların en büyük zorluğu başlıklarda fikir ve duygunun aynı anda bulunuyor olmasıdır. Başlık öncelikli, model tabanlı çalışmaların en bilinen uygulamalarından birisi literatürde kısaca LDA olarak geçen (uzun hali Latent Dirichlet Allocation) ve Türkçeye gizli Dirichlet ayrımı olarak çevrilebilecek olan yöntemdir. Bu yaklaşımda metinler gizli başlıkların karışımlarını istatistiksel bir modele oturtulmaktadır ve bu model metinlerdeki kelime dağılımlarından çıkarılmaktadır [11].

Örneğin aşağıdaki üç cümleyi ele alalım:

Ben **balık** ve sebze yerim

Balıklar evcil hayvanlardır

Benim kedim **balık** yer

Yukarıdaki üç cümlede iki farklı başlıkta fikir çıkarımı yapılabilir. İlk başlık "yemek" olarak ve ikinci başlık "evcil hayvan" olarak çıkarıldıktan sonra, yukarıdaki cümlelerde kalın harflerle yazılı olanların "yemek" başlığında ve yattık olarak yazılanların ise "evcil hayvan" başlığındaki kelimeler olduğu söylenebilir. Buna göre birinci cümle

%100 "yemek" başlığında, ikinci cümle %100 "evcil hayvan" başlığındayken üçüncü cümle %33 "evcil hayvan" ve %66 "yemek" balığı altında kabul edilmelidir.

Bu örnekte görüldüğü gibi LDA iki aşamadan oluşmaktadır, öncelikle metinlerden başlıkların çıkarılması ve ardından da başlıklara göre metinlerin sınıflandırılması. Bu iki aşamadan ikincisi yani metinlerin başlıklara atanması da iki alt aşamadan oluşmaktadır. İlk alt aşama metindeki her kelimenin geçici olarak bir başlığa atanması ardından metindeki kelimelerin yoğunluğuna göre metnin bir başlığa atanması. Ancak bazı durumlarda bir kelime birden fazla başlığa ait olabilir, bu yüzden LDA tekrarlı şekilde (iterative) doğru başlığı bulmak için istatistiksel modelini güncellemektedir. Örneğin yukarıdaki üç cümlede geçen "balık" kelimesi bir evcil hayvan veya bir yemek olarak kabul edilebilir. Biz doğal dili kullanan insanlar olarak "benim kedim balık yer" cümlesini okuyunca buradaki "balık" kelimesinin bir yemek olduğunu anlayabiliyoruz ancak bilgisayarın bu analizi yapabilmesi için "yemek" fiilinin balığı işaret ettiğini çözümlenmesi gerekir. Doğal dil analizi yapılmadan sadece istatistiksel olarak analiz yapılan durumlarda bunun anlaşılması biraz karmaşık ve vakit alıcı olabilir. Basitçe elimizdeki bir sözlükle kelimeleri arayarak bu kelimelerin dahil olacağı sınıfları bulmak aslında çoğu zaman yanlış sonuçlar doğurmaktadır. Bu yüzden ihtimal olarak bir kelimenin birden fazla sınıfa farklı oranlarda dahil olduğu kabulü yapılmakta ve her tekrarda (iterasyon) daha doğru sınıfa atama yapılmaktadır. Örneğin Türkçede balık kelimesi daha yüksek ihtimalle yiyecek olarak kullanılmakta (diyelim ki %80) ve daha düşük oranda bir evcil hayvan olarak kullanılmaktaysa LDA öncelikle kelimeyi bir yemek olarak sınıflandıracak ancak daha sonra cümledeki diğer kelimelerin kullanımına göre sınıfını değiştirerek evcil hayvan sınıfına alabilecektir (veya tam tersi).

1.4. Kanaat Çıkarımında Eş Görüş Kümelemesi

Kanaat çıkarımı (fikir madenciliği) çalışmalarının önemli bir kısmı da şimdiye kadar alt bölümler halinde anlatılan bakış tabanlı, frekans tabanlı veya model tabanlı yaklaşımlar ile çizge teoreminin birlikte kullanılmasından doğar. Buna göre bir sosyal ağda bulunan kişilerin fikirlerinin birbirini etkilemesi veya diğer bir deyişle sosyal ağda fikir yayılımı mümkündür. Aynı fikre sahip kullanıcıların aynı kümede toplandığı ve sosyal ağın aynı fikre sahip kişiler olarak kümelendiği yaklaşımlara eş görüş kümelemesi (homophily) yaklaşımı denilmektedir [12]*. Eş görüş çalışmalarını temel kabul ederek farklı amaçlar için kullanan çalışmalar da vardır. Örneğin bir sosyal ağdaki yapının eş demografik özelliklere göre kümelenmesi de mümkündür. Bu çalışmaların amacı bir sosyal ağı, fikir madenciliği yöntemleri ile yaş gruplarına göre veya cinsiyete göre kümelemek olarak görülebilir [13].

Her ne kadar fikir yayılması ve sosyal ağın farklı şekillerde kümelenmesi çalışmalarında fikir madenciliği teknikleri kullanılsa da problemin en önemli farklılığı hareketli bir ortamda çalışıyor olmasıdır. Örneğin bir kişinin yaptığı basit bir paylaşımla başlayan süreç, çok kısa sürede yüzbinlerce kişiye ulaşmakta ve etkileme süreci başlamaktadır. Bu kişilerden kaçının etkilendiği, kaçının bu fikri beğendiği ve yaymaya başladığının bulunması için problemin hareketli (dinamik) bir yapıda ele alınması gerekir [14].

Problemin dinamik yapısından kaynaklanan özelliği düşünüldüğünde fikir özetleme problemi de farklı bir boyut kazanır. Örneğin bir metinde fikir özetlemesi yapıldıktan sonra elde edilen çok sayıdaki fikrin sadece bir kısmının yayıldığı görülebilir. Bu durumda fikir özetlemesi sadece yayılan fikirlere odaklanarak diğer fikirleri eleme yoluna gidebilir.

---

* Literatürdeki "homophily" kelimesinin tam çevirimi, "homo" eş/benzer ve "phily" sevmek anlamında olduğu için "benzer sevgi" veya "eş sevgi" şeklinde yapılabilir ancak kavramı tam karşılaması açısından "eş görüş" ifadesinin daha uygun olacağı düşünülerek bu şekilde kullanılmıştır.

2. Duygu Analizinde Karşılaşılan bazı zorluklar

Duygu analizi doğal dilde yazılmış metinleri işlediği için doğal dilden kaynaklanan bazı problemleri miras almıştır. Bunun yanında enformasyon seviyesindeki problemler de eklenmiştir. Bazı güncel problemler aşağıdaki şekilde sıralanabilir:

2.1. Kapalı Duygu ve Alaycılık (Implicit Sentiment and Sarcasm)

Bazı cümlelerde "ima" yoluyla doğrudan olmayan ve kapalı ifadeler bulunabilir.

"Birisi bu filmi oturup nasıl izler?"

şeklindeki bir soru aslında cevap aranan bir sorudan ziyade olumsuz duygu belirten (negative sentiment) bir cümle olarak algılanmalıdır. Benzer bir örnek aşağıdaki şekilde verilebilir:

"Bu kitabı yazan kişinin akli dengesini sorgulamak gerekir".

Yukarıdaki örneklerde de görüldüğü gibi genelde kapalı duygu ifade eden ve alaycı cümleler geniş zaman, emir kipi ve soru ifadeleri şeklinde geçmektedir.

2.2. Alan Bağlılığı (Domain Dependency)

Aynı kelimelerin farklı alanlarda farklı anlamlara geldiği görülmektedir. Örneğin aşağıdaki iki cümle için aynı kelime farklı anlamlara gelmektedir:

"Filimin gidişi tahmin edilemez şekilde akıyor"

"Arabanın gidişi tahmin edilemez şekilde akıyor"

İlk örnekte bir filim için tahmin edilemez olmak olumlu bir özellik iken ikinci örnekte yer alan araba için tahmin edilemez olmak olumsuz bir özelliktir. Benzer şekilde "kitabı oku" şeklinde bir emir cümlesi bahsi geçen kitap için olumlu ve tavsiye edici bir emir olurken, bazı alanlarda bilgisizliği vurgulayan ve bilgi eksikliğine dikkat çeken olumsuz bir cümle olabilmektedir.

2.3. Ters Kabuller (Thwarted Exceptions)

Bazan yazar tarafından bir argüman sistemi kurularak bütün kurulan argüman sisteminin tersini iddia eden ve doğrulayan tek bir cümle ile kabulün terse çevrildiği görülmektedir. Aşağıda bu duruma bir örnek sunulmuştur.

"Çok parlak bir filimdi. Filimin kurgusu oldukça başarılı ve oyuncu seçimi tam yerindeydi, Stallone oldukça iyi bir performans çıkarmıştı. Ancak bütün bunlar yetersiz kalmaktaydı".

Yukarıdaki cümleler teker teker ele alındığında oldukça olumlu cümlelerin sonunda tek bir olumsuz cümle bulunmakta ve bu olumsuz cümle bütün olumlu cümlelerin etkisini negatif yönde etkilemektedir.

Klasik duygu analizi yaklaşımlarında bu metnin terim frekansı, veya terim belirleme yöntemlerine göre başarısız sonuç vereceği ve metnin olumlu duygu olarak sınıflandırılacağı söylenebilir.

2.4. Pargmatik (Pragmatics)

Bazı cümlelerin duygu analizi sırasında kullanılan ifadeler sonraki cümlelerin duygu analizini etkilemektedir. Örnek olarak İngiliz İngilizcesini içeren aşağıdaki tabloda benzer vurgular yapılmıştır:

WHAT THE BRITISH SAY	WHAT THE BRITISH MEAN	WHAT FOREIGNERS UNDERSTAND
I hear what you say	I disagree and do not want to discuss it further	He accepts my point of view

With the greatest respect	You are an idiot	He is listening to me
That's not bad	That's good	That's poor
That is a very brave proposal	You are insane	He thinks I have courage
Quite good	A bit disappointing	Quite good
I would suggest	Do it or be prepared to justify yourself	Think about the idea, but do what you like
Oh, incidentally/ by the way	The primary purpose of our discussion is	That is not very important
I was a bit disappointed that	I am annoyed that	It doesn't really matter
Very interesting	That is clearly nonsense	They are impressed
I'll bear it in mind	I've forgotten it already	They will probably do it
I'm sure it's my fault	It's your fault	Why do they think it was their fault?
You must come for dinner	It's not an invitation, I'm just being polite	I will get an invitation soon
I almost agree	I don't agree at all	He's not far from agreement
I only have a few minor comments	Please rewrite completely	He has found a few typos
Could we consider some other options	I don't like your idea	They have not yet decided

*With thanks to the (unknown) author of this table, first posted by Duncan Green of Oxfam.

Yukarıdaki tabloda vurgulanan İngilizce örneklerine benzer örnekler Türkçe için de verilebilir. Kelimelerin ve cümlelerin ifadesinden çok terimlerin ifadesine dönüşen bu yapılardan bazıları aşağıda sıralanmıştır:

"Ne var, ne yok"; "X ne yaşar ne yaşamaz" v.s.

2.5. Dünya Bilgisi (Knowledge)

Her ne kadar duygu analizi enformasyon seviyesinde bir problem olsa da dünya bilgisine sahip bazı vurguların yakalanması da söz konusu olabilir. Örneğin aşağıda bazı cümlelerde bilgi ihtiyacı duyulmaktadır:

"Şeytan gibi futbolcu"

"Çocuklar duymasın cebe girdi"

Yukarıdaki cümle örneklerinde bazı isimlerin daha önceden biliniyor olması ve bu isimlerin ifade ettiği duygu dünyasının anlaşılıyor olması gerekir. Örneğin "Azrail", "şeytan", "frenkeştayn" gibi kelimelerin ifade ettiği anlamlar çok kolay şekilde bu amaçla kullanılabilir. Veya güncel dizi, filim, roman isimleri, güncel kısaltma ve anlamlar biliniyor olmalıdır. Örneğin "cebe girmek" ile ifade edilen bilgi aslında cep telefonunu işaret etmektedir.

2.6. Öznellik Yakalanması (Subjectivity Detection)

Bu aşamada yapılan çalışmalarda metinde ifade edilen vurgunun kişisel ve hatta kişiliğin eleştirisini mi yoksa nesnenin ve duygusallığı araştırılan varlığın eleştirisi mi olduğunu ayırmayı amaçlar. Örneğin,

"Aşk filimlerinden nefret ederim"

Cümlesi, bir filimi eleştirmek ve hatta negatif veya pozitif kutupsallık ifade etmekten ziyade kişinin kendisi ile ilgili yaptığı bir öz eleştiridir.

2.7. Varlık Tanımlama (Entity Identification)

Bir cümlede bahsi geçen birden fazla varlık (entity) olabilir ve bu varlıklarla ilgili olumlu/olumsuz duygu analizi farklılıklar gösterebilir. Aşağıdaki örnekleri ele alırsak:

"Samsung,Nokia'dan daha iyi"

"Microsoft'un uzun dönemli yatırım politikası, Linux'u geçmek için yetersiz kalacak gibi görülüyor".

Yukarıdaki iki örnekte de aynı cümlede ikişer varlık ismi geçmiş ve birisi hakkında olumlu duygu bulunurken diğeri hakkında olumsuz duygu bulunmaktadır. Örneğin ilk cümlede Samsung için olumlu olan ifade, Nokia için olumsuzluk ifade etmektedir.

Klasik frekans ve terim çıkarımı yöntemlerinin bu tip cümlelerde başarılı olmayacağı anlaşılmaktadır.

2.8. Ters İfadeler (Negation)

Duygu analizindeki bir diğer problem ise olumsuzluk eklerinin etki alanlarının bulunmasıdır. Örneğin aşağıda iki farklı cümlede olumsuzluk ekleri kullanılmıştır:

"Okula gelmemiş olman, kalacağın anlamına gelmez"

"Okula gelmedin ama yine de kalmayacaksın"

"Okula geldin ama yine de kalacaksın"

Yukarıdaki örneklerde görüldüğü üzere, kelimenin olumsuz ek alması ve "ama" bağlacının olumsuzluğu birleşerek olumlu sonuç doğurmaktadır. Bununla birlikte olumsuzluk eki ve "ama" bağlacı tek başın cümlenin duygusal analizini ters yönde değiştirmektedir. Ayrıca olumsuzluk ekinin etkisi bağlaca kadar devam etmektedir.

3. Duygu Analizinde veri ön işleme ve özellik çıkarım yöntemleri

Bu bölümde, proje kapsamında kullanılması olası özellik çıkarım yöntemleri ve bu yöntemlerin teknolojik yansımaları incelenmiştir.

3.1. TF-IDF

TF-IDF kavramı IR (information retrieval, bilgi getirimi) gibi konuların altında bir sıralama (ranking) algoritması olarak sıkça geçmektedir.

İngilizcedeki Term Frequency – Inverse Document Frequency (Terim frekansı – ters metin frekansı) olarak geçen kelimelerin baş harflerinden oluşan terim basitçe bir metinde geçen terimlerin çıkarılması ve bu terimlerin geçtiği miktara göre çeşitli hesapların yapılması üzerine kuruludur.

Klasik olarak TF yani terimlerin kaç kere geçtiğinden daha iyi sonuç verir. Kısaca TF-IDF hesabı sırasında iki kritik sayı bulunmaktadır. Bunlardan birincisi o anda ele alınan dokümandaki terimin sayısı diğeri ise bu terimi külliyatta içeren toplam doküman sayısıdır.

Örnek:

Konuyu bir örnek üzerinden açıklayalım:

Örneğin 100 dokümandan oluşan bir külliyatımız olsun ve TF-IDF hesaplamak istediğimiz kelime de "şadi" olsun. Bu durumda birinci dokümana bakıp "şadi" kelimesinin kaç kere geçtiğini sayarız. Diyeli ki 4 kere geçiyor olsun. Ardından külliyatımızdaki 100 dokümandan kaçında "şadi" kelimesi geçiyor diye bakarız. Diyelim ki 10 dokümanda bu kelime geçiyor olsun (dikkat edilecek husus kelimenin geçip geçmediğidir diğer dokümanlarda kaç kere geçtiğinin bir önemi yoktur).

Şimdi TF ve IDF değerlerini ayrı ayrı hesaplayacağız ve sonra bu iki değeri çarpacağız, önce TF hesabına bakalım:

TF hesabı için ihtiyacımız olan bir diğer değer ise o andaki dokümanda en fazla geçen terim sayısıdır. Örneğin o anda baktığımız dokümanda en fazla geçen terimimizin sayısı da 80 olsun.

İlk hesaplama dokümanda bizim ilgilendiğimiz kelimenin en fazla geçen kelimeye oranıdır. Yani kelimemiz 4 kere geçtiğine ve en fazla geçen kelimemiz de 80 kere geçtiğine göre ilk oranımız (ki bu oran aynı zamanda TF (term frequency, terim frekansı) olarak tek başına da anlamlıdır)

TF = 4 / 80 = 0,05 olarak bulunur.

Ardından IDF değerini hesaplayalım. Bunun için basit bir bölme işlemi yapılacak ve logaritması alınacaktır.

$$IDF(t,D) = log\frac{Toplam\ Doküman\ Sayısı}{Terimi\ içeren\ doküman\ Sayısı} = \log\frac{|D|+1}{DF(t,D)+1}$$

Buna göre IDF için toplam 100 dokümandan 10 dokümanda aradığımız kelime "şadi" geçtiğine göre

IDF = log (100 / 10) = log (10) = 1 olarak bulunacaktır.

IDF hesabı sırasında bir iki noktaya dikkat etmek gerekir. Öncelikle logaritmanın tabanının bir önemi yoktur. Amaç üstel fonksiyonun tersi yönde bir hesap yapmaktır. Doğal logaritma kökü e, 2 veya 10 gibi sayılar en çok kullanılan değerlerdir. Genelde TF-IDF değerinin kıyas için kullanıldığını ve diğer terimlerin TFIDF değerleri ile kıyaslandığını düşünecek olursak hepsinde aynı tabanın kullanılıyor olması sonucu değiştirmeyecektir.

Diğer dikkat edilecek bir husus ise IDF hesabı sırasında geçen "terimi içeren doküman sayısı" değeridir. Bu değer hesaplama sırasında paydada yer almaktadır ve bu değerin 0 (sıfır) olma ihtimali vardır. Bu durumda sonuç sıfıra bölüm belirsizliğine götürebileceğinden genelde bu değere 1 eklemek sıkça yapılan bir programlama yaklaşımıdır.

Neticede elde ettiğimiz TF = 0,05 ve IDF = 1 değerlerini çarpıyoruz ve terimimizin TF-IDF değeri aşağıdaki şekilde bulunuyor:

TF-IDF = TF x IDF = 0,05 x 1 = 0,05

Yukarıda kullandığımız formülleri aşağıdaki şekilde de açıkça yazmak mümkündür:

$$w_{i,d} = tf_{i,d} \times \log\frac{n}{df_i}$$

Yukarıdaki gösterimde, i terimi için ve d dokümanı için hesaplama yapılmaktadır. Öncelikle TF hesaplanır ki bu basitçe terimin o dokümanda kaç kere geçtiğinin en fazla geçen terime oranı şeklinde hesaplanabilir:

$$tf_{i,d} = \max\frac{fr_{i,d}}{df_i}$$

Yani i terimi için d dokümanındaki terim frekansı (term frequency), i teriminin d dokümanındaki tekrar sayısının o dokümandaki en yüksek tekrar sayısına sahip terimin tekrar sayısına oranıdır. Veya bu oranların en yükseğidir.

Yukarıda verilen TF-IDF formülünde ayrıca n toplam doküman sayısını df ise doküman frekansını vermektedir ve df aslında i teriminin kaç farklı dokümanda geçtiğinin sayısıdır.

Son olarak TF-IDF yönteminin diğer yöntemlere göre farkını açıklamaya çalışalım. TF-IDF ile bir terimin kaç kere geçtiği kadar kaç farklı dokümanda da geçtiği önem kazanır. Örneğin sadece bir dokümanda 100 kere geçen bir terimle 10 farklı dokümanda onar kere geçen terimin ikisi de aslında toplamda 100 kere geçmiştir ancak TF-IDF ikincisine yani daha fazla dokümanda geçene önem verir.

Spark ortamında, TF-IDF hesaplaması için MLib üzerinde hazır olarak sunulan kütüphaneler bulunmaktadır. Spark üzerinde TF-IDF için kullanılan kütüphanede karım hilesi (hashing trick) olarak geçen yöntem ile arama işlemleri bir karım fonksiyonuna indirgenmekte ve çalışma hızında etkili şekilde artış olmaktadır. Karım hilesini de içeren TF-IDF koduna ait örnek aşağıda sunulmuştur:

```
import org.apache.spark.rdd.RDD
import org.apache.spark.SparkContext
import org.apache.spark.mllib.feature.HashingTF
import org.apache.spark.mllib.linalg.Vector

val sc: SparkContext = ...

// Her satırda tek bir doküman yüklenerek
val documents: RDD[Seq[String]] = sc.textFile("...").map(_.split(" ").toSeq)

val hashingTF = new HashingTF()
val tf: RDD[Vector] = hasingTF.transform(documents)
```

Yukarıda sunulan kodda TF değeri (veya Spark isimlendirmesine göre HashingTF değeri) sadece tek bir geçiş gerektirirken IDF hesaplaması sırasında dokümanların üzerinden iki geçiş gerekmektedir.

```
import org.apache.spark.mllib.feature.IDF

// ... önceki örnekten devam…
tf.cache()
val idf = new IDF().fit(tf)
val tfidf: RDD[Vector] = idf.transform(tf)
```

3.2. Kelime Vektör Dönüşümü (Word2Vec)

Kelimelerin analizi sırasında kullanılan yöntemlerden birisi de kelimelerin bir vektör olarak ifade edilmesidir. Klasik olarak 3 farklı dokümandan oluşan bir örnekte kelimelerin vektör dönüşümleri aşağıdaki şekilde yapılabilir:

- *John likes to watch movies.*
- *Mary likes movies too.*
- *John also likes football.*

Terim	Indis
John	1
likes	2
to	3
watch	4
movies	5
Mary	6
too	7
also	8
football	9

Yukarıda verilen terim indekslerinin birer vektöre dönüşmüş hali aşağıdaki şekilde sunulmuştur.

$$\begin{pmatrix} \text{John} & \text{likes} & \text{to} & \text{watch} & \text{movies} & \text{Mary} & \text{too} & \text{also} & \text{football} \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Yukarıdaki örnekte de görüldüğü üzere artık her dokümanı ikilik tabanda bir vektör ile ifade etmek mümkündür. Örneğin ilk doküman “11110000” olarak ifade edilebilir.

Ancak bu ifade yöntemi klasik metin işleme problemlerinde başarılı sonuçlar verirken özellikle dağıtık çalışmayı amaçlayan ve Map-Reduce şeklinde dağıtık çalışacak olan ortamlarda bütün metnin tek bir ortamda bulunması ve diğer metinlerde geçen kelimelerin tam listesinin gerekmesi gibi uygulama güçlükleri doğurmaktadır.

Çözüm olarak skip-gram olarak geçen yöntem önerilmektedir. Skip-gram yönteminde kelimelerin arka arkaya gelen sıklıklarının analiz edilmesi yerine kelime analizlerinin belirli aralıklara indirgenmesi ve bu analiz grupları arasında boşlukların olmasına imkan verilmiştir.

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{j=-k}^{j=k}\log p(w_{t+j}|w_t)$$

Yukarıda verilen denklemde k değeri kayan pencere yaklaşımındaki pencere boyutunu belirtmektedir. Buna göre skip-gram pencere boyutu kadar kelimeyi alarak bu kelimeleri işlemektedir. Skip-gram yöntemi ayrıca her kelime için iki ayrı vektörde güncelleme yapmaktadır. İlk olarak kelimelerin sayılarının tutulduğu u vektörü ve ardından da kelimelerin geçtiği bağlam vektörü v güncellenmektedir.

Ardından bu iki vektördeki göreceli olarak koşullu olasılık değerlerinin güncellenmesi aşağıdaki denklemde gösterildiği şekilde yapılmaktadır:

$$p(w_i|w_j) = \frac{\exp\,(u_{w_i}^T v_{w_j})}{\sum_{l=1}^{w}\exp\,(u_l^T v_{w_j})}$$

Yukarıdaki denklemde geçen V, kelime hazinesinde bulunan kelimelerin sayısını ifade etmektedir.

Yukarıdaki denklemde verilen koşullu olasılık değerinin hesaplanması yöntemin yazılıma geçirildiği aşamada işlem maliyeti olarak karşımıza çıkmaktadır. Bunu azaltmak için hiyerarşik softmax yöntemi kullanılmakta ve işlemin karmaşası O(log(V)) seviyesine indirilmektedir.

```
import org.apache.spark._
import org.apache.spark.rdd._
import org.apache.spark.SparkContext._
import org.apache.spark.mllib.feature.Word2Vec

val input = sc.textFile("text8").map(line => line.split(" ").toSeq)

val word2vec = new Word2Vec()

val model = word2vec.fit(input)

val synonyms = model.findSynonyms("china", 40)

for((synonym, cosineSimilarity) <- synonyms) {
  println(s"$synonym $cosineSimilarity")
}
```

3.3. Standart Ölçekleme (StandardScaler)

Kullanılabilecek diğer bir özellik çıkarım yöntemi de ölçekleme etkisinin özelliklerin dağılımı üzerine uygulanmasıdır. Örneğin dağılım oranları farklılık gösteren farklı veri kümelerinin ortak ölçeklere indirgenmesi için dağılımların ortalama değerleri (mean) ve standart sapmalarının (veya varyanslarının) ortak bir dağılıma indirgenmesi gerekmektedir. Bu işlem için standart ölçekleme yöntemi kullanılabilir. Ölçekleme sırasında ayrıca ortalama değerlerinin eşitlenmesi isteniyorsa parametrik olarak ayarlanma imkanı da bulunmaktadır. Aksi halde farklı ortalama değerlerinde (örneğin boşluk yoğun girdilerde (sparse input)) ölçekleme yapılabilmektedir. Yöntem, almış olduğu girdi vektörünü ölçekleyerek sonuç üretmektedir. Yöntemin kullanıma dair örnek kod aşağıda verilmiştir:

```
import org.apache.spark.SparkContext._
import org.apache.spark.mllib.feature.StandardScaler
import org.apache.spark.mllib.linalg.Vectors
```

```
import org.apache.spark.mllib.util.MLUtils

val data = MLUtils.loadLibSVMFile(sc, "data/mllib/sample_libsvm_data.txt")

val scaler1 = new StandardScaler().fit(data.map(x => x.features))
val scaler2 = new StandardScaler(withMean = true, withStd = true).fit(data.map(x => x.features))
// scaler3 is an identical model to scaler2, and will produce identical transformations
val scaler3 = new StandardScalerModel(scaler2.std, scaler2.mean)

// data1 will be unit variance.
val data1 = data.map(x => (x.label, scaler1.transform(x.features)))

// Without converting the features into dense vectors, transformation with zero mean will raise
// exception on sparse vector.
// data2 will be unit variance and zero mean.
val data2 = data.map(x => (x.label, scaler2.transform(Vectors.dense(x.features.toArray))))
```

3.4. Normalleştirme (normalization)

Ön işleme sırasında kullanılan ölçekleme yöntemine benzer şekilde normalleştirme yaklaşımı da veri kümelerinin ortak bir yapıya indirgenmesini amaçlar.

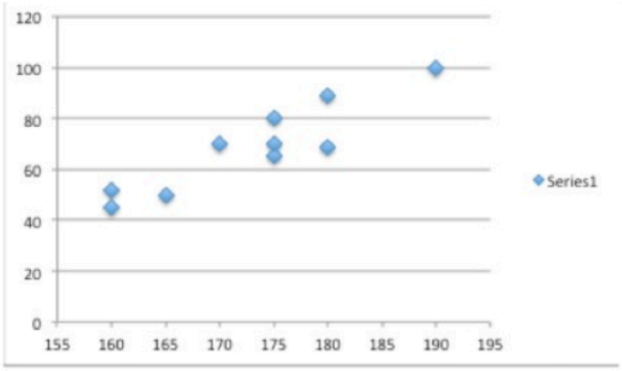
Örneğin TF-IDF özellik çıkarım yöntemi ile çıkarılan iki özellik vektörü arasındaki benzerlik için kosinüs benzerliğinin kullanılacağını düşünelim. Bu durumda vektörlerdeki özellikler arasında bütün özelliklerin karşılaştırılarak hesaplandığı bir süreç başlayacaktır. Bunun yerine özellik vektörlerinin karesi alınarak (L2) bu iki özellik vektörünün klasik çarpma yöntemi ile (dot product) çarpılması sonucunda kosinüs benzerliği bulunabilir. Normalleştirme aşamasında vektörün istenen bir norm değerle normalize edilmesi mümkündür ve yöntem bu değeri p parametresi olarak alır. Varsayılan değer olarak p = 2 kabul edilir.

```
import org.apache.spark.SparkContext._
import org.apache.spark.mllib.feature.Normalizer
import org.apache.spark.mllib.linalg.Vectors
import org.apache.spark.mllib.util.MLUtils

val data = MLUtils.loadLibSVMFile(sc, "data/mllib/sample_libsvm_data.txt")

val normalizer1 = new Normalizer()
val normalizer2 = new Normalizer(p = Double.PositiveInfinity)

// Each sample in data1 will be normalized using $L^2$ norm.
val data1 = data.map(x => (x.label, normalizer1.transform(x.features)))

// Each sample in data2 will be normalized using $L^\infty$ norm.
val data2 = data.map(x => (x.label, normalizer2.transform(x.features)))
```

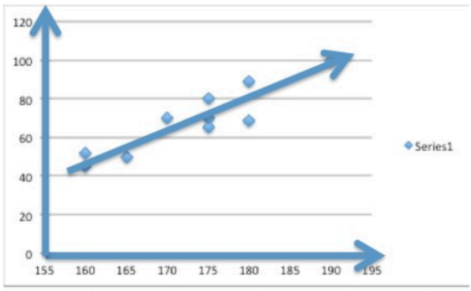
3.5. PCA (Principal Component Analysis)

Birincil veya esas bileşen analizi olarak da bilinen Principal Component Analiysis-PCA, veri madenciliğinde veri analiz edildiğinde veri üzerindeki bileşenlerini çıkartmaya yarayan bir filtreleme çeşitidir. PCA sayesinde veri üzerinde yeni boyutlar elde edilir. Bu sayede kullanılan algoritmanın başarım oranı artmaktadır. Bir örnek üzerine konuşulması gerekirse bir sınıf üzerindeki bir kişinin boy ve kilo bilgisine dayanarak sınıfının tahmin edilmesidir. Örneğin 1.70 boyu ve 90 kilo ağırlığındaki bir kişiye tecrübelere dayanarak erkek olduğu kanısına varılabilir. Fakat burada PCA ile bu sınıftaki kişilerin özelliklerini iki boyut üzerinden yani Boy-Kilo ile yeni iki boyut üzerinden düşünülecektir. Bu sayede daha doğru bir sınıflandırma yapılması hedeflenmektedir.

Boy	Kilo	Cinsiyet
160	52	k
175	80	e
165	50	k
160	45	k
180	89	e
175	65	k
175	70	e
180	69	k
190	100	e
170	70	k



Sınıf örneğindeki veriler bir excel sayfasında oluşturulup bir grafik elde edildiğinde verilerin bazılarının birbirine yakın olması beklenir. Bu durumda doğru sınıflandırma için öyle bir eğri çizilmelidir ki yeni ve belirsiz bir veri girildiğinde doğru bir sınıflandırma elde yapılabilsin. Fakat verilerin birbirine çok yakın olması durumunda bu tahminin çok da kolay olmayacaktır. PCA sayesinde bu özellikler ortaya çıkarılarak veri üzerinde yeni boyutlar elde edilir.



MLib kütüphanesinde bulunan PCA aşağıdaki örnek kodda gösterildiği gibi çalıştırılabilir

```
import org.apache.spark.mllib.regression.LinearRegressionWithSGD
import org.apache.spark.mllib.regression.LabeledPoint
import org.apache.spark.mllib.linalg.Vectors
import org.apache.spark.mllib.feature.PCA

val data = sc.textFile("data/mllib/ridge-data/lpsa.data").map { line =>
  val parts = line.split(',')
  LabeledPoint(parts(0).toDouble, Vectors.dense(parts(1).split(' ').map(_.toDouble)))
}.cache()

val splits = data.randomSplit(Array(0.6, 0.4), seed = 11L)
val training = splits(0).cache()
val test = splits(1)

val pca = new PCA(training.first().features.size/2).fit(data.map(_.features))
val training_pca = training.map(p => p.copy(features = pca.transform(p.features)))
val test_pca = test.map(p => p.copy(features = pca.transform(p.features)))

val numIterations = 100
val model = LinearRegressionWithSGD.train(training, numIterations)
val model_pca = LinearRegressionWithSGD.train(training_pca, numIterations)

val valuesAndPreds = test.map { point =>
  val score = model.predict(point.features)
```

```
  (score, point.label)
}

val valuesAndPreds_pca = test_pca.map { point =>
  val score = model_pca.predict(point.features)
  (score, point.label)
}

val MSE = valuesAndPreds.map{case(v, p) => math.pow((v - p), 2)}.mean()
val MSE_pca = valuesAndPreds_pca.map{case(v, p) => math.pow((v - p), 2)}.mean()

println("Mean Squared Error = " + MSE)
                        println("PCA Mean Squared Error = " + MSE_pca)
```

4. Duygu Analizinde Makine Öğrenmesi algoritmaları

Duygu analizi için sık kullanılan yaklaşımlardan bazıları proje kapsamında araştırılmıştır. Raporda bahsi geçen makine öğrenmesi ve özellik çıkarım yöntemleri kullanılarak duygu analizini hedef alan çalışmalar ve bu çalışmalar için kullanılan algoritmalar bu bölümde açıklanacaktır.

Genel olarak duygu analizi için kullanılan iki temel yol bulunmaktadır [15]. Bunlardan ilki doğal dil işleme yöntemlerini kullanarak ilgili doğal dil (Türkçe, İngilizce gibi) için özel olarak geliştirilmiş şekilbilimsel ve sözdizimsel analizlerin ardından anlambilimsel sonuçlara ulaşmak ve ulaşılan sonuçlara göre duygu analizi yapılması mümkün olmaktadır [16]. İkinci bir yol ise istatistiksel yöntemler kullanmak ve metin üzerinden istatistiksel özellik çıkarımı yaparak elde edilen sayısal değerlerin karar vermeye destek için kullanılmasıdır [15].

İki yöntemde de kesin başarı garantisi olmamakla birlikte doğal dil işleme yaklaşımının görece olarak daha yüksek başarı oranına sahip olduğu, bununla birlikte istatistiksel yaklaşımların görece düşük başarı oranı yanında daha hızlı çalıştığı görülmektedir.

Duygu analizi çalışmalarının uygulama alanına göre bu iki yaklaşım arasında seçim yapılmaktadır. Örneğin, sosyal ağlar gibi akan veri üzerinde gerçek zamanlı ve zaman kısıtlaması olarak yapılan çalışmalarda hız önceliğinden dolayı istatistiksel modellerin kullanıldığı yaklaşımlar daha çok tercih edilmektedir.

Klasik olarak kullanılan n-gram [17] veya entropi [18] çıkarımı gibi yöntemler kullanılarak istatistiksel modeller ile çıkarım yapılarak sonuçlar yorumlanmaktadır.

İki yaklaşım için de işlem yapılan dilin sonuca etkisi olduğu görülmüştür. Özellikle İngilizce gibi üzerinde çok fazla çalışma yapılan dillerdeki başarı oranlarının zaman içerisinde daha yüksek oranlara çıktığı ancak Türkçe gibi görece olarak az sayıda çalışma yapılan dillerde başarı oranının daha düşük kaldığı anlaşılmaktadır. Bu raporda da bahsedilmiş olan duygu analizi problemleri göz önüne alındığında insanların duygu analizini %85 oranlarında başarı ile yapabildiği anlaşılmaktadır. Şu anda bilgisayar yazılımları için hedef bu başarı oranını yakalamaktır ve ancak %70 oranındaki doğru sınıflandırma başarılı olarak kabul edilmektedir [19].

Örneğin sadece twitter verileri üzerinde yoğunlaşılmış ve İngilizce metinleri işleyen bir çalışmada başarı oranı (f1-score) %76 - %88 arasında değişmektedir [20]. Farklı bir çalışmada özellik çıkarım yöntemleri üzerine yoğunlaşılmış ve kelimelerin köklerine ayrılması (stemming), n-gram, terim frekansı (TF) gibi yöntemler kullanılarak yapılan bir araştırmada farklı veri kümeleri üzerinde %86 ile %97 arasında başarı (f1-score) elde edilmiştir [21].

Türkçe için yapılan çalışmalarda ise başarı oranı sinema yorumları için %83 oranında başarı elde etmiş çalışmalar bulunmaktadır [22]. İngilizce duygu analizi sözlüğünün birebire tercümesi ile elde edilen sözlüğün uygulanması sonucunda haber verileri üzerinde yapılan bir analizde %74 oranında ortalama başarı (accuracy) sonucuna ulaşılmıştır [23].

Yapılan literatür araştırması sonucunda farklı özellik çıkarım yöntemlerine, farklı makine öğrenmesi algoritmalarına ve literatürde elde edilen sonuçların başarı oranlarına ulaşılmış ve bu bilgiler üzerinden mevcut proje için farklı duygu analizi imkanlarına ulaşılmıştır. Projenin bir sonraki iş paketinde ulaşılan bu duygu analizi yöntemleri kullanılarak literatürde bulunan başarı oranlarının üzerinde başarı oranlarına ulaşılması hedeflenmektedir.

5. Referanslar

[1] Sadi Evren Seker, "Event Ordering for Turkish Natural Language Texts," in *CSW-2010 1 st Computer Science Student Workshop*, Istanbul, 2010, pp. 26-29.

[2] Sadi Evren Seker, C Mert, K Al-Naami, N Ozalp, and U Ayan, "Time Series Analysis on Stock Market for Text Mining Correlation of Economy News," *International Journal of Social Sciences and Humanity Studies*, vol. 6, no. 1, pp. 69 - 91, Mar. 2014.

[3] Mohammad M. Masud, Latifur Khan, and Ehab Al-Shaer, "Email Worm Detection Using Naïve Bayes and Support Vector Machine," *Intelligence and Security Informatics*, vol. 3975, pp. 733-734, 2006.

[4] Bing Liu, "Chapter 11, Opinion Mining and Sentiment Analysis," in *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*.: Springer, 2011, pp. 459-526.

[5] Minqing Hu and Bing Liu, "Mining and summarizing customer reviews," *Proceeding KDD '04 Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining* , pp. 168-177 , 2004.

[6] Ana-Maria Popescu and Oren Etzioni, "Extracting product features and opinions from reviews ," *Proceeding HLT '05 Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing* , pp. 339-346 , 2005.

[7] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani, "Multi-facet rating of product reviews ," *Proceeding ECIR '09 Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval* , pp. 461 - 472 , 2009.

[8] Samaneh Moghaddam and Martin Ester, "Opinion digger: an unsupervised opinion miner from unstructured product reviews," *Proceeding CIKM '10 Proceedings of the 19th ACM international conference on Information and knowledge management* , pp. 1825-1828 , 2010.

[9] Himabindu Lakkaraju, Chiranjib Bhattacharyya, Indrajit Bhattacharya, and Srujana Merugu, "Exploiting Coherence for the Simultaneous Discovery of Latent Facets and associated Sentiments," *SIAM International Conference on Data Mining (SDM)*, pp. 498-509, 2011.

[10] Tak-Lam Wong, Wai Lam, and Tik-Shun Wong, "An unsupervised framework for extracting and normalizing product attributes from multiple web sites," *Proceeding SIGIR '08 Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* , pp. 35-42 , 2008.

[11] David M. Blei, Andrew Y. Ng, and Michael I. Jordan, "Latent Dirichlet Allocation," *Journal of Machine Learning Research*, pp. 993-1022, 2003.

[12] McPherson, Smith-Lovin, L. bM., and J. M Cook, "Birds of a feather: Homophily in social networks ," *Annual review of sociology* , pp. 415-444, 2001.

[13] M. O Jackson, *Social and economic networks*.: Princeton University Press , 2010.

[14] M. Kaschesky, P. Sobkowicz, and G Bouchard, "Opinion Mining in Social network: Modelling, Simulating, and Visualizing Political Opinion Formation in the Web ," *The Proceedings of 12th Annual International Conference on Digital Government Research* , 2011.

[15] Hinrich Schuetze Christopher Manning, *Foundations of Statistical Natural Language Processing*. USA: MIT Press, 1999.

[16] James H. Martin Daniel Jurafsky, *Speech and Language Processing*.: Prentice Hall, 2008.

[17] C. Y. Suen, "n-Gram Statistics for Natural Language Understanding and Text Processing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1, no. 2, pp. 164 - 172, 1979.

[18] Vincent J. Della Pietra , Stephen A. Della Pietra Adam L. Berger, "A maximum entropy approach to natural language processing," *Journal of Computational Linguistics* , vol. 22, no. 1, pp. 39-71 , 1996.

[19] John Burn-Murdoch, "Social media analytics: are we nearly there yet?," *The Guardian: Big data The data store: on big data*, June 2013.

[20] Furu Wei‡ , Nan Yang♮, Ming Zhou‡, Ting Liu†, Bing Qin† Duyu Tang†, "Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification," *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pp. 1555-1565, 2014.

[21] Padmini Srinivasan Yelena Mejova, "Exploring Feature Definition and Selection for Sentiment Classifiers ," *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*, pp. 546-549, 2011.

[22] Alaettin Uçan, Ebru Akcapinar Sezer and Hayri Sever Fırat Akba, "ASSESSMENT OF FEATURE SELECTION METRICS FOR SENTIMENT ANALYSES: TURKISH MOVIE REVIEWS," *European Conference Data Mining 2014 and International Conferences Intelligent Systems and Agents 2014 and Theory and Practice in Modern Computing 2014*, pp. 180-184, 2014.

[23] Cigdem Aytekin, "An Opinion Mining Task in Turkish Language: A Model for Assigning Opinions in Turkish Blogs to the Polarities," *Journalism and Mass Communication*, vol. 3, no. 3, pp. 179-198, 2013.

[24] Reichheld & Sasser, "Zero Defections: Quality comes to service," *Harvard Business review.*, 1990.

[25] G Fripp, "Guide to CLV, Guide to Customer Lifetime Value," 2014.