T.C.
PAMUKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

BÜYÜK VERİ ARAÇLARINDAN HADOOP KULLANARAK VERİ MADENCİLİĞİ

YÜKSEK LİSANS TEZİ

MEHMET UMUT SALUR

DENİZLİ, KASIM - 2016

T.C.
PAMUKKALE ÜNIVERSITESI
FEN BILIMLERI ENSTITÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

BÜYÜK VERİ ARAÇLARINDAN HADOOP KULLANARAK VERİ MADENCİLİĞİ

YÜKSEK LİSANS TEZİ

MEHMET UMUT SALUR

DENİZLİ, KASIM - 2016

KABUL VE ONAY SAYFASI

Mehmet Umut SALUR tarafından hazırlanan "**Büyük Veri Araçlarından Hadoop Kullanarak Veri Madenciliği**" adlı tez çalışmasının savunma sınavı 28.11.2016 tarihinde yapılmış olup aşağıda verilen jüri tarafından oy birliği / oy çokluğu ile Pamukkale Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı Yüksek Lisans Tezi olarak kabul edilmiştir.

Jüri Üyeleri	İmza
Danışman Doç. Dr. Sezai TOKAT	.....
Üye Yrd. Doç. Dr. Meriç ÇETİN Pamukkale Üniversitesi	.....
Üye Yrd. Doç. Dr. İbrahim Berkan AYDİLEK Harran Üniversitesi	.....

Pamukkale Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 02/12/2016 tarih ve 44/19 sayılı kararıyla onaylanmıştır.

.....

Prof. Dr. Uğur YÜCEL

Fen Bilimleri Enstitüsü Müdürü

Bu tezin tasarımı, hazırlanması, yürütülmesi, araştırmalarının yapılması ve bulgularının analizlerinde bilimsel etiğe ve akademik kurallara özenle riayet edildiğini; bu çalışmanın doğrudan birincil ürünü olmayan bulguların, verilerin ve materyallerin bilimsel etiğe uygun olarak kaynak gösterildiğini ve alıntı yapılan çalışmalara atfedildiğine beyan ederim.

Mehmet Umut SALUR

ÖZET

BÜYÜK VERİ ARAÇLARINDAN HADOOP KULLANARAK VERİ MADENCİLİĞİ
YÜKSEK LİSANS TEZİ
MEHMET UMUT SALUR
PAMUKKALE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ
BILGISAYAR MÜHENDISLIĞI ANABILIM DALI
(TEZ DANIŞMANI:DOÇ. DR. SEZAİ TOKAT)

DENİZLİ, KASIM - 2016

Bu tez çalışması kapsamında günümüzde çok önemli bir konu haline gelen büyük verinin işlenmesi incelenmiştir. Büyük veriden anlamlı bilgiler çıkarmak günümüz hem özel sektör hem de kamu alanı için önemli bir konudur. Bu nedenle birçok kuruluş bu alan için ciddi yatırımlar yapmaktadır. Büyük veri genellikle yapısal olmayan verilerden oluşmaktadır. Yapısal olmayan verilerden anlamlı bilgiler elde etmekte doğal dil işleme yöntemleri kullanılmaktadır. Doğal dil işleme yöntemlerini kullanarak duygu analizi yapmak birçok alanda önemli avantajlar sağlamaktadır. Bu tez çalışması kapsamında büyük veri işleme araçlarından olan Hadoop üzerinde veri madenciliği yöntemleriyle duygu analizi yapılması hedeflenmiştir.

Veri madenciliği kapsamında metin madenciliği kullanılmıştır. Hadoop üzerinde veri madenciliği yapmak için özelleştirilmiş olan Mahout aracı kullanılmıştır. Mahout makine öğrenmesi algoritmalarının map-reduce formatında yazılmış hallerini içeren bir kütüphanedir. Metin madenciliğinde kullanılan veri kümesi için Türkiye'deki 15 günlük gazetenin Twitter 'da paylaşmış oldukları haber başlıkları kullanılmıştır. Bu haber başlıkları Türkçe doğal dil işleme için geliştirilen Zemberek kütüphanesi yardımıyla ön işlemlerden geçirilmiştir. Bu haber başlıkları olumlu veya olumsuz olarak sınıflandırılmıştır. Sınıflandırma işlemi için Mahout aracıyla birlikte Naive Bayes istatistik tabanlı sınıflandırma algoritması kullanılmıştır. Sınıflandırma işleminden önce Naive Bayes algoritması için eğitim verisi oluşturulmuştur. Eğitim verisi için yaklaşık 105.000 haber başlığı, yazılan bir uygulama yardımıyla kullanıcı tarafından olumlu, olumsuz veya belirsiz olarak işaretlenmiştir.

Eğitim verisinin bir kısmı algoritmanın eğitilmesi, bir kısmı ise algoritmanın testi için kullanılmıştır. Naive bayes algoritmasının çalıştırılması için iki farklı Hadoop ortamı oluşturulmuş. Bu ortamlar tek node'luk Hadoop sistemi ve 4 node'luk Hadoop sistemi şeklindedir. Sınıflandırma işlemi her iki ortamda gerçekleştirilmiştir. Sınıflandırma işleminde %80'e yakın başarı elde edilmiştir.

ANAHTAR KELİMELER: Hadoop, Mahout, Veri Madenciliği, Büyük Veri, Duygu Analizi

ABSTRACT

DATA MINING USING HADOOP BIG DATA TOOL
MSC THESIS
MEHMET UMUT SALUR
PAMUKKALE UNIVERSITY INSTITUTE OF SCIENCE
COMPUTER ENGINEERING
(SUPERVISOR:ASSOC. PROF. DR. SEZAİ TOKAT)

DENİZLİ, NOVEMBER 2016

This thesis has investigated the big data which has become very popular topic in recent days. Drawing meaningful information from big data is an important topic for both private and public sectors. Thus, many companies have made serious investments. The big data is made of unstructured data. The natural language process methods have been used to obtain meaningful information from unstructured data. Using the natural language process methods to analyze emotion is bringing important advantages to many fields. His study aimed to analyze emotion with data mining method by using Hadoop which is a tool for big data processing.

The text mining is used within data mining process. The Mahout tool which is specialized to do data mining on Hadoop is used. Mahout is a library that contains machine learning algorithms its map-reduce formats. The data set used for the text mining has been drawn from the headlines of 15 Turkish daily newspapers Twitter posts. This headlines has been filtered with Zemberek library developed for natural language process of Turkish. These headlines were classified as positive and negative. Mahout and Naive Bayes statistical based classification algorithms tools are used for the classification. The learning data is prepared for the Naive Bayes algorithm prior to the classification process. More than 105 thousands headlines are drawn from twitter with a developed software for the learning data and this data is marked as positive, negative, and uncertain.

The part of the learning data has been used for the learning algorithm and the other part has been used for the testing of the algorithm. The type of Hadoop environment was developed in order to run the Naive Bayes algorithm. These environments were single node Hadoop system and 4-node Hadoop system. The classification process has been carried out in the both systems. In the classification, success was achieved close to 80%.

KEYWORDS: Hadoop, Mahout, Data Mining, Big Data, Sentiment Analysis

İÇİNDEKİLER

Sayfa

ŞEKİL LİSTESİ

Sayfa

TABLO LİSTESİ

Sayfa

SEMBOL LİSTESİ

HDFS	**:**	Hadoop Distributed File System
TDK	**:**	Türk Dil Kurumu
TBL	**:**	Transformation Based Learning
SMO	**:**	Sequential Minimal Optimization
ASF	**:**	Apache Software Foundation
GSM	**:**	Global System for Mobile Communications
API	**:**	Application Programming Interface
RSS	**:**	Really Simple Syndication
HTML	**:**	Hypertext Markup Language
DBSCAN	**:**	Density-Based Spatial Clustering of Applications with Noise
IDC	**:**	International Data Corporation
GFS	**:**	Google File System
SQL	**:**	Structured Query Language
RDD	**:**	Resilient Distributed Dataset
KDD	**:**	Knowledge Discovery in Databases
XML	**:**	Extensible Markup Language
TF	**:**	Term Frequency
IDF	**:**	Inverse Document Frequency
SSD	**:**	Solid-State Disk
SSH	**:**	Secure Shell

ÖNSÖZ

Hayatım boyunca bana destek olan aileme, tez sürecinde göstermiş oldukları sabır ve desteklerinden dolayı sonsuz teşekkürlerimi sunarım. Bu tez çalışması boyunca destek ve yardımlarını esirgemeyen danışman hocam, Sayın Doç. Dr. Sezai TOKAT'a çok teşekkür ederim. Ayrıca tez çalışması sürecinde tecrübelerinden faydalandığım hocam Yrd. Doç. Dr. İbrahim Berkan AYDİLEK'e teşekkürü bir borç bilirim. Hem bu tez çalışmasında hem de akademik çalışmalarda desteklerini esirgemeyen, her soruma cevap vermeye çalışan Yrd. Doç. Dr. Gürhan GÜNDÜZ'e de teşekkürlerimi sunuyorum.

1. GİRİŞ

Her gün teknolojik gelişmelere dair yüzlerce yeni haber yapılmakta, bu haberlerin birçoğunda yeni bir ürün veya bir üründeki geliştirmelerden bahsedilmektedir. Ürün, yeni bir yazılım veya fiziksel bir teknolojik ürün olabilmektedir. Bu baş döndüren teknolojik gelişim sürecinin toplum üzerinde bazı çıktıları mevcuttur. Bu çıktılardan biri, insanların artık her dakika her saniye dijital veri üretmeleridir. Kişisel bloglar, telefonlar, marketlerdeki kasalar, otobüs kartları, uçak seferleri, oyunlar, sosyal medya, kamu kuruluşları ve daha birçok alanda çok ciddi bir hızda her gün yeni veri üretilmektedir.

İnsanlık tarihi boyunca üretilen verilerin saklanması önem arz etmiştir. Yakın tarihten günümüze veriler kâğıt kalemden sonra gelişen teknoloji yardımıyla bilgisayarda dijital olarak saklanmıştır. Bilgisayar ortamında bu veriler başta dosyalarda saklanmış, dosyalardan sonra daha verimli çözümler sunan veri tabanı yönetim sistemlerinde saklanmaya başlanmıştır. Saklanan bu verilerden anlamlı sonuçlar çıkarmak amacıyla bu veriler işlenmeye başlanmıştır. Günümüzden geçmişe, yaklaşık son 10 yıllık süreçte üretilen bu veri çok büyük bir ivme kazanmıştır. Bu verilerin saklanmasında kullanılan teknolojilerin veya yazılımların yetersizliği söz konusu olmaya başladı. Bunun nedenlerinden biride üretilen verinin çok farklı formatlarda, özelliklerde olmasıdır. Metin dosyalarının yanında resim, ses, video, sayısal veriler gibi birçok veri türü üretilmeye başlanmıştır.

Üretilen verilerin çok değişik formatlarda olması ve üretim hızının çok yüksek olması yeni kavram ve teknolojilerin ortaya çıkmasına neden olmuştur. Bu kavramlardan biri "Büyük Veri" olmuştur. Büyük veri, bir veri kümesiyle ilgili normal veri tabanı yönetim sistemlerinde saklanan verilerin yanında yapısal olamayan verilerinde bir arada olması, bu verilerin yüksek boyutlarda(terabayt, petabayt) olması ve aynı zamanda çok hızlı büyümeye devam etmesi durumunu ifade eder. Bu büyüme hızına dair olarak; 2020 yılında dijital olarak üretilecek verinin 2009 yılında üretilen dijital verinin yaklaşık olarak 44 katı olacağı öngörülmektedir (Big Data Statistics, 2016).

Üretilen bu devasa veriyi yönetmek, saklamak ve işleyebilmek için günümüzde birçok teknolojik çözüm üretilmekte, geliştirilmektedir. Bu teknolojilerden biri de Hadoop teknolojisidir (Wikipedia Hadoop, 2016). Hadoop düşük maliyetli donanım, yüksek hesaplama kapasitesi, yüksek performans ve dağıtık hesaplama ilkelerini gerçekleştiren bir büyük veri çözümüdür. Hadoop yardımıyla çok büyük boyuttaki veriler üzerinde işlem yapılabilmektedir. Hadoop kendisine özgü HDFS(Hadoop Distributed File System) dosya sistemini kullanmaktadır. Bu özelliğinin yanında map-reduce programlama tekniğini kullanmaktadır. Hadoop yardımıyla veriler ile ilgili bir hesaplama, bir veri sorgusu veya veriden anlamlı bilgi çıkarma işlemleri yapılabilmektedir.

Veriden anlamlı çıkarımlar yapmak, fark edilmeyen davranışların keşfedilmesi, öngörülemeyen sonuçların tespiti günümüzde önemli bir konudur. Bu çalışmalar disiplin olarak veri madenciliği çalışma alanına girmektedir. Veri madenciliği bir veri kümesine dair normal tekniklerle(istatiksel, veri tabanı sorgusu vb.) öngörülemeyen, cevaplandırılamayan soruların cevaplarını, çeşitli algoritmaların yardımıyla ortaya çıkaran bir çalışma alanıdır. Üretilen bu büyük verinin içinden öngörülemeyen bilgilerin keşfedilmesi günümüzde her alanda büyük bir öneme sahiptir. Hastanelerdeki hasta kayıtlarından hastalık teşhisine, bankalardaki sahtekârlık tespitinde, müşteri memnuniyet analizinde, müşteri tüketim davranışların tespitinde, ürün birliktelik analizinde, öğrenci notlarına göre başarı analizlerinde, log analizlerinden web sunucularına saldırı tespiti gibi birçok alanda veri madenciliği disiplini uygulama alanı bulmuştur. Veri madenciliğinde verilerin toplanması, temizlenmesi, uygun formata dönüştürülmesi ve uygun algoritmanın çalıştırılması gibi süreçleri vardır.

Veri madenciliğinin bir alt başlığı olaraktan görülebilen metin madenciliği birkaç basamağı dışında veri madenciliği sürecinin metin dosyaları üzerindeki gerçekleştirimidir. Veri madenciliğinde veriler genellikle veri tabanı yönetim sistemlerinden veya veri ambarlarından temin edildiğinden veriler yapısal türdendir. Metin madenciliğinde ise veriler yapısal değildir. Metin madenciliğindeki veriler sosyal bloglardaki metinler, bir web sitesinin içeriği vb. olabilmektedir. Bu yapısal olmayan veriler üzerinde veri madenciliği yöntemlerinin uygulanması için öncelikle bu verilerin yapısal formata dönüştürülmeleri gerekmektedir. Metin madenciliği

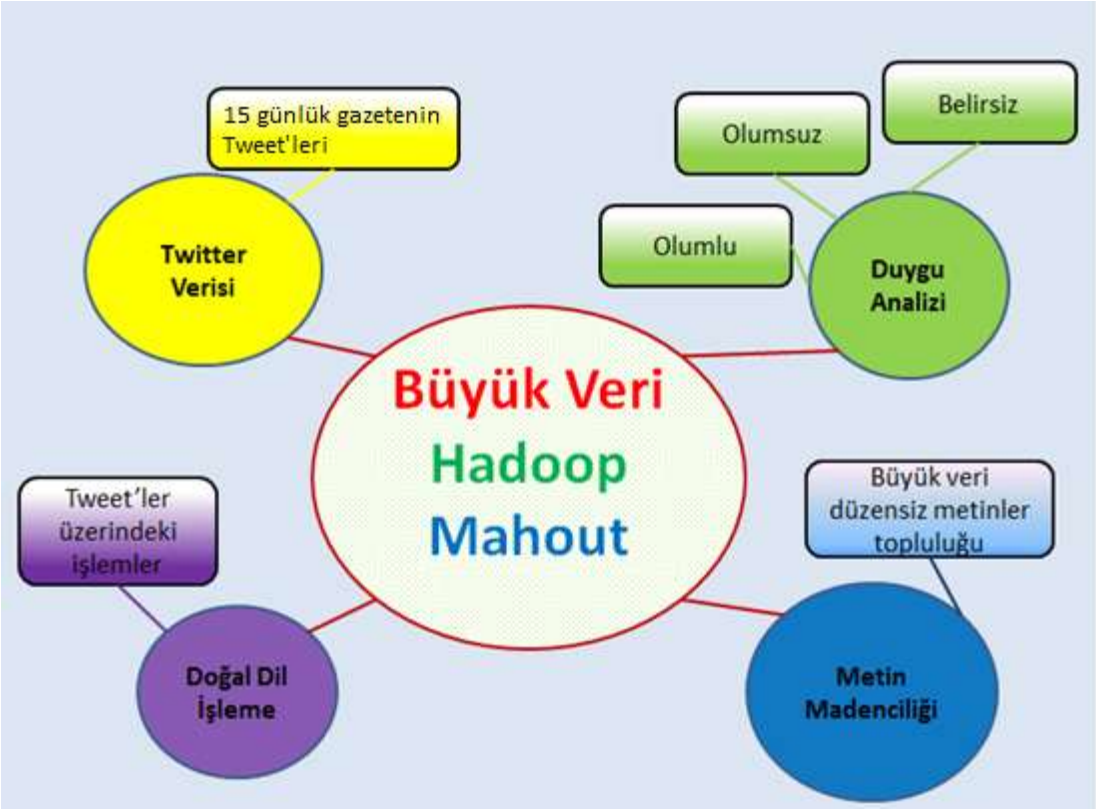
sürecinde gerek ön işlemlerde gerekse de bu dönüşüm için doğal dil işleme yöntemleri uygulanmaktadır. Doğal dil işleme yöntemiyle cümledeki kelimelerin yapısı belirlenebilmekte, cümle üzerindeki morfolojik analizler yapılabilmektedir. Doğal dil işleme yöntemi günümüzde popüler bir disiplin haline gelmiş bulunmaktadır. Doğal dil işleme yöntemiyle otomatik cevaplama sistemleri, otomatik metin düzeltme/tamamlama, duygu analizi gibi birçok uygulaması mevcuttur.

Hızlı bir büyüme eğrisine sahip büyük veri üzerinde veri madenciliği yöntemlerinin uygulanması ve anlamlı sonuçların çıkarılması bu tez çalışmasının temel amacıdır. Bu amaç doğrultusunda altyapı olarak güçlü yapısıyla popüler olan büyük veri araçlarından Hadoop tercih edilmiştir. Hadoop üzerinde büyük verinin bir parçası olan sosyal medyadan alınan veriler üzerinde çalışılmıştır. Bu veriler Twitter sosyal paylaşım bloğundan alınmıştır. Türkiye'deki 15 günlük gazetenin Twitter hesaplarından alınmış olan tweet'ler üzerinde çalışılmıştır. Gazeteler günlük olarak gelişen haberleri dünyaya servis etmektedirler. Gazeteler tarafından Twitter'da haberin başlığı veya haberin özetinden sonra haberin linkini içeren tweet'ler atmaktadırlar. Yazılmış olan bir uygulama aracılığıyla Twitter'dan atılan bu tweet'ler toplanmıştır. Uygulama periyodik olarak farklı zamanlarda çalıştırılmış ve yaklaşık 105.000 civarında tweet toplanmış ve dosya olarak kayıt edilmiştir. Kayıt edilen bu verilerin Hadoop yardımıyla iki kategori altında sınıflandırılması amaçlanmıştır.

Sınıflandırma işlemi için alınan cümleler üzerinde doğal dil işleme yöntemleri uygulanmıştır. Türkçe dilinde doğal dil işlemi için yazılmış olan açık kaynak kodlu Zemberek (Zemberek, 2016) kütüphanesi kullanılmıştır. Ön işlemlerden sonra veriler Hadoop üzerinde çalışan ve veri madenciliği algoritmalarının map-reduce yöntemiyle gerçekleştirilmiş halini içeren Mahout framework'ü kullanılmıştır. Mahout'un sağlamış olduğu metin madenciliği algoritmalarından Naive Bayes algoritması tercih edilmiştir.

Naive Bayes algoritması için yazılmış olan "Eğitim Verisi Hazırlama" uygulamasıyla, atılmış olan tweet'ler elle olumlu, olumsuz ve belirsiz olarak işaretlenmiştir. Oluşturulan eğitim verisiyle algoritma eğitilmiş ve tweet'ler olumlu veya olumsuz olarak sınıflandırılmıştır.

Sınıflandırma işlemi için iki farklı altyapı oluşturulmuş, bunlardan birincisi tek node'luk hadoop sanal bilgisayarıdır, diğeri ise 4 node'luk hadoop dağıtık küme sistemidir. Kurulumu gerçekleştirilen hadoop bilgisayarları için sanal bilgisayar oluşturma işlemine olanak sağlayan Oracle VM VirtualBox (VirtualBox , 2016) programı kullanılmıştır. Hadoop kütüphanesi Ubuntu işletim sistemi üzerinde çalıştırılmıştır. Sınıflandırma işlemi her iki bilgisayar sistemi üzerinde çalıştırılmış ve önemli istatistikler tezin 6'ncı bölümünde paylaşılmıştır. Tez çalışmasının bileşenleri Şekil 1.1'de gösterildiği gibidir.



Şekil 1.1: Tez özet görünümü

2. LİTERATÜR BİLGİSİ

2.1 Doğal Dil İşleme

Tümce içindeki hemen hemen her sözcüğün birden fazla anlamı mevcuttur. İnsan beyni tümce içindeki sözcüğün konuya veya anlama en çok bütünlük sağlayan anlamını seçmektedir. Altan ve Orhan çalışmalarında tümce içindeki bir sözcüğün belirsizliğini üzerinde durmuşlardır (Altan ve Orhan 2005). Bu konuda yapılan çalışmalardan ve projelerden bahsetmişlerdir. Hesaplamalı dil bilim çalışmaları için bu konunun önemi üzerinde durmuşlardır.

Doğal dil işleme günümüzde birçok alanda yer edinmiş önem arz eden bir çalışma alanıdır. Başlıca konuşma analizi, konuşma tanımlama ve imla doğrulama alanlarında çalışılmaktadır. Aktaş çalışmasında Türkçe metin dosyalarında, metni cümlelere ayırma problemine çözüm olarak geliştirilen bir yöntemden bahsetmiştir (Aktaş 2006). Metni cümlelere ayırma işlemi için bir kural tablosu(Türkçe 'de hangi durumlarda cümle bitiş karakterleri (".", "…","?") kullanıldığına dair kurallar) oluşturulmuş ve bu kural tablosu geliştirilmiş olan uygulama yüklenmekte, sistem çıktı olarak xml formatında metnin cümlelere ayrılmış halini kullanıcıya vermektedir.

Aşlıyan ve diğerleri çalışmalarında Türkçe 'deki kelimelerin doğru bir şekilde hecelere ayrılması amaçlanmıştır(Aşlıyan ve diğ. 2006). Türkçe 'deki hece yapısını ifade eden bir model referans alınarak, kullanılmış olan Türkçe külliyatı veri kümesindeki kelimeler % 100'e yakın bir başarı oranıyla hecelere ayrılmıştır. Kelimeleri doğru bir şekilde hecelere ayıran algoritma kendileri tarafından geliştirilmiştir.

Güngör ve diğerleri çalışmalarında bir sözlükte var olan sözcükler arasındaki anlamsal bağlantıları ve ilişkileri ortaya koyan ve kurallara bağlı hiyerarşik bir sözcükler yapısı oluşturmayı amaçlayan bir yöntem geliştirilmişlerdir(Güngör ve Güngör 2007). Üzerinde çalıştıkları metin tek tek kelimelere ayrılmış, metindeki isim ve sıfatlar için eş anlamlı ve üst-kavram çalışması yapılmış, yapılan çalışmada her kelimenin Türkçe sözlükteki açıklamasına bir algoritma uygulayarak üst-kavram ve eş anlamlı çıkarımı yapılmıştır. Daha sonra bir sözcük için belirlenen üst-kavram seçme

kriteri analizi uygulanmıştır. Türkçe dili için, Türk Dil Kurumu'nun(TDK) sunmuş olduğu elektronik sözlük kullanılarak, sözcükler arasında bazı ilişkileri ortaya koymuşlardır. Bu ilişkiler üst kavram, alt-kavram ve eş anlamlılık bağlantılarıdır. Yöntemlerinde bu ilişkileri otomatik bir şekilde tespit etmeyi hedefleyen bir yöntem sunmuşlardır.

Doğal dil işlemenin bir diğer farklı uygulamasını Dikici ve Saraçlar gerçekleştirmişlerdir(Dikici ve Saraçlar 2008). Bu çalışmada kayan alt yazıların tanınması amaçlanmıştır. Çalışmada öncelikle çeşitli videolar örneklem olarak alınmış ve bu örnekler üzerinde görüntü işleme yardımıyla kayan yazıdaki karakterler tanınmaya çalışılmıştır. Karakter tanıma işlemi doğal dil işleme alanında çok sık bir şekilde kullanılan bir denetimli sınıflandırma yöntemi olan Dönüşüm Tabanlı Öğrenme (TBL – Transformation Based Learning) yöntemi yardımıyla yapılmıştır. Dönüşüm tabanlı öğrenme algoritması için eğitim verisi olarak haber sitesinden toplanan veriler kullanılmıştır. Bu veri kümesi yaklaşık olarak 11 milyon karakterden oluşmaktadır.

Özkaya ve Diri doğal dil işlemenin bir alt başlığı olan varlık isimi tanıma üzerine bir sistem gerçekleştirmişlerdir(Özkaya ve Diri 2011). Sistemlerinde gayri resmi bir dille yazılmış olan e-postalar arasından Şartlı Rastgele Alanlar kullanılarak kurum-kuruluş, yer, özel isimlerin etiketlendikten sonra çıkarılması için etkin bir sistem geliştirmişlerdir. Çalışma için Java programlama diliyle geliştirilmiş olan bir kütüphane kullanılmıştır. Çalışmada veri seti olarak 150 adet e-posta kullanılmıştır(50 kurumsal, 50 kişisel, 50 akademik). Kısaltma isimleri belirlemek için 175 adet kısaltma listesi kullanılmıştır. Uygulamada kurumsal bilgiler içeren e-postalar en yüksek başarıyı göstermiş olup, ortalamada %87 oranında doğru tanıma yapılmıştır. Ayrıca minimum hata ile belirlenen varlık ismi %95 oranıyla kişi isimleri olarak sonuçlanmıştır.

Doğal dil işleme disiplinin birçok uygulama alanı mevcuttur. Otomatik metin düzeltme, söz ve yazım yanlışları tespiti, dilin hangi alanda iyi kullanıldığı vb. geniş bir çalışma alanı vardır. Bu alanların yanı sıra doğal dil işleme disiplininde en çok ilgi gören konu duygu analizi olmuştur. Bir metinin ifade ettiği anlamın bilgisayar tarafından anlaşılması, birçok alanda büyük gelişmelere neden olacaktır.

2.2 Duygu Analizi

Teknolojinin hızlı gelişmesi sonucu olarak internet erişimi çok geniş bir alanda sağlanmaktadır. İnternetin sağlamış olduğu avantajlardan biri çevirim içi pazar alanı sunmasıdır. Bu pazarlarda müşteri memnuniyetini en hızlı şekilde sağlamak günümüz firmaları için çok büyük önem arz etmektedir. Çevirim içi olarak hizmet veya ürün alan müşteriler, ürün veya hizmet ile ilgili düşüncelerini çevirim içi sosyal medya veya blog ortamlarında dile getirmektedirler.

Pazar alanında üstünlük sağlamak ve kar amacını iyileştirmek isteyen firmalar, müşterilerin sosyal medya veya diğer paylaşım platformlarında kendileri hakkında yazmış oldukları yorumları değerlendirmek çok önemlidir. Bu değerlendirmelerin sistematik olarak hızlı ve kolay sağlanması şüphesiz bu yorumların doğru analiz edilmesini gerektirmektedir. Bu değerlendirilmelerin yapılması noktasında duygu analizi çalışma alanı olarak karşımıza çıkmaktadır.

Duygu analizi temel olarak bir metin içinde geçen bir yargının sınıflandırılması işlemidir. Metinlerdeki bu yargılar en çok olumlu, olumsuz veya nötr sınıflara ayrılmak istenmektedir. Metinlerin sınıflandırılması sonucu firmalar müşterileriyle daha hızlı ve olumlu sonuçlar doğurabilecek etkileşimlere girebilmektedirler. Duygu analiziyle ilgili literatürde yapılan bazı çalışmalardan aşağıdaki başlık altında bahsedilmiştir.

2.2.1 Türkçe için Yapılan Çalışmalar

Türkiye de yapılan duygu analizi çalışmalarından ilki olarak kabul görebilen Eroğlu (2009), yaptığı çalışmada duygu analizinde kullanılabilecek iki yeni veri seti oluşturmuş ve İngilizce dili için yapılan çalışmaları Türkçe dili için göstermiş olduğu başarı değerlendirilmiştir. Çalışmasında film yorumlarını olumsuz veya olumlu olarak sınıflandırmıştır.

Çakmak ve diğerleri çalışmalarında Türkçe için cümle ve kelime bazlı olarak duygu analizi yapmışlardır (Çakmak ve diğ. 2012). Analiz için 31 adet çocuk kitabından alınan 83,120 cümleden oluşan veri kümesi üzerinde çalışmışlardır. Kelime

kökeninin anlamı ile cümlenin bir bütün olarak anlam ilişkisine dair bazı istisnalar hariç yüksek paralellik olduğu bulunmuştur.

Boynukalın öncelikle İngilizce bir veri kümesini elle Türkçe 'ye çevirip bir veri kümesi oluşturmuştur (Boynukalın 2012). Bu veri kümesine bir takım Türkçe verileri de elle ekledikten sonra oluşturmuş olduğu veri kümesi üzerinde duygu analizi yapmıştır. Yaptığı analizde makine öğrenmesi teknikleriyle veri kümesini sınıflandırmıştır. Sınıflandırma işlemlerini sonucunda veri kümesini %80 oranında doğru sınıflandırmıştır.

Şimşek ve Özdemir 6 aylık bir süreçte Türkiye'den atılan yaklaşık 1.9 milyon tweet'i kullanarak çalışmalarında, ekonomi ile borsa değişimi arasında bir ilişkinin varlığı üzerinde çalışmışlardır (Şimşek ve Özdemir 2012). Duyguları temel olarak 8 ayrı sınıfa ayırmışlardır, bu sınıfları ifade edecek 113 anahtar kelime seçmişlerdir. Bunun sonucunda tweet'leri mutlu ve mutsuz olmak üzere iki sınıfa ayırmışlardır. Sonuç olarak borsa değişimi ve atılan tweet'ler arasında % 45 oranında bir ilişkinin olduğunu saptamışlardır.

Akbaş tezinde Türkçe tweet'ler üzerinde konu temelli duygu analizi yapmıştır (Akbaş 2012). Öncelikle elle (manuel olarak) duygu kelime listesi oluşturulmuş, toplanan tweet'ler arasından kelime seçme algoritması yardımıyla belirlenmiş olan duygu kelimelerin makine öğrenmesi algoritmaları yardımıyla üretilmesi amaçlanmıştır.

Duygu analizinde farklı bir alan olarak siyasi haberler üzerinde bir çalışma yapmışlardır (Kaya ve diğ. 2012). Çalışmalarında dört farklı denetimli makine öğrenmesi yöntemiyle haber sitelerinden almış oldukları haberleri farklı öznitelikler ve ağırlıklandırma yöntemleri kullanılarak sınıflandırılmıştır. 4 algoritmanın oranları karşılaştırılmış ayrıca Hürriyet gazetesinin İngilizce haberleri ve Türkçe haberleri de karşılaştırılmıştır.

Çetin ve Amasyalı çalışmalarında Telekom sektöründeki iki şirketin tweet'lerini (her şirketin 6000 örneği) kullanmışlardır(Çetin ve Amasyalı 2013). Cümle analizinde kelime kökleri ve karakter gramları kullanılmıştır. Tweet'ler olumlu, olumsuz ve nötr üç ayrı sınıfta tutulmuştur. Çalışmada farklı algoritmaların

performansları karşılaştırılmış ve metin sınıflandırmada kullanılan eğitici ve eğiticisiz terim ağırlıklandırma yöntemlerinin karşılaştırmalı analizi yapılmıştır.

Kaya ve diğerleri çalışmalarında politik köşe yazarlarının hesaplarından alınan iki Twitter veri kümesi ve özel oluşturulmuş olan bir diğer veri kümesi üzerinde duygu sınıflandırması yapılmıştır(Kaya ve diğ. 2013). Bu çalışmada sınıflandırma performansını arttırmak amacıyla veri etiketleme işleminde fayda sağlayan bilgi aktarımı(transfer learning) kullanılmıştır. Bu yöntem sayesinde sınıflandırma başarımı yaklaşık olarak %23 oranında artmıştır. Sınıflandırma işlemi için çeşitli makine öğrenmesi yöntemleri kullanılmıştır.

Meral ve Diri çalışmalarında 9 farklı alandan toplam 8321 adet tweet toplamış, bu tweet'ler üzerinde Türkçe için geliştirilmiş olan doğal dil işleme kütüphanesi kullanılmıştır(Meral ve Diri 2014). Bu kütüphane zemberek ismindedir, zemberek ile kelimeler hecelere ayrılmış(2-gram ve 3-gram) sonra bu veriler üzerinde Rastgele Orman, Naive Bayes ve Destek Vektör Makinesi gibi önemli makine öğrenmesi algoritmaları kullanılarak tweet'ler sınıflandırılmıştır. N-gram modelinde en yüksek başarımı 0.90 f-ölçüm değeriyle Rastgele Orman algoritması sağlamıştır.

Akba ve diğerleri çalışmalarında 5662 film hakkındaki 219198 yorumu iki farklı özellik seçimi yöntemi kullanarak, bu yorumları denetimsiz öğrenme algoritması olan karar destek vektör makinesi yardımıyla sınıflandırmışlardır(Akba ve diğ. 2014). Film yorumlarını iki sınıfa(olumlu-olumsuz) ayırdıklarında yaklaşık %84, üç sınıfa ayırdıklarında ise yaklaşık % 63 başarım elde etmişlerdir.

Nizam ve Akın çalışmalarında gıda firmalarının farklı ürünlerinin yorumlarından oluşan Twitter veri seti üzerinde denetimli öğrenme yaklaşımı kullanılarak duygu analizi çalışması yapmışlardır(Nizam ve Akın 2014). Veri seti üzerinde farklı sınıflandırma algoritmaları kullanılarak başarımları incelenmiştir. En başarımlı olan algoritma SMO(Sequential Minimal Optimization) olmuştur, yaklaşık %72 oranında başarım göstermiştir.

Beyhan tezinde Twitter'dan otomatik olarak alan ve sınıflandıran bir özel yazılım yardımıyla, ülkemizde iletişim hizmeti veren üç GSM(Global System for Mobile Communications) operatörün (Avea, Turkcel, Vodafone) isimlerinin anahtar

sözcük olarak kullanılarak atılmış olan tweet'ler üzerinde metin madenciliği yöntemleriyle duygu analizi ve kümeleme analizini yapmıştır(Beyhan 2014). Duygu analizi yapılarak cümleleri olumlu, olumsuz veya nötr olarak sınıflandırmıştır. Sınıflandırılmış olan bu veriler üzerinde bazı işlemler gerçekleştirilmiş ve kümeleme analizi yapılması amacıyla birleştirilmiştir. Bazı doğal dil işleme adımları uygulandıktan sonra 2, 3, 4, 5, 6, 7 ve 8'li kümelere bölünmüş ve en iyi performansı sağlayan küme sayısı tespit edildikten sonra duygu analizi orijinal verisiyle veri kümesiyle birleştirilmiştir. Her kümeye ait duygu dağılımı gözlemlenmiştir. Örneğin romantik kümesindeki verilerin üç operatör için olumlu olumsuz ve belirsiz tweet sayılarını grafiksel olarak göstermiştir. Ayrıca GSM operatörlerine ait ilginç sonuçlar paylaşılmıştır.

Uçan tezinde, duygu analizinde kullanılan makine öğrenmesi algoritmalarının yerine, İngilizce için oluşturulmuş olan bir duygu sözlüğünü Türkçe' ye çevirmiş ve doğrudan sözlük kullanarak duygu analizin yapılmasına olanak sağlayacak Türkçe Duygu Sözlüğü oluşturmayı amaçlamıştır(Uçan 2014). Sözlük çeviri işlemi için birden fazla çevirici sistemden(Google Translate API, tureng.com, zargan.com) faydalanılmıştır. Sözlüğün doğru duygu analizi yaptığını, film ve otel yorumlarından oluşan veri kümesine makine öğrenmesi yöntemleri uygulanmış analizler yapılmış ve sonuçların başarılı olduğu paylaşılmıştır.

2.3 Metin Madenciliği

Metin madenciliği çalışmaları yaklaşık olarak 1980 yıllarında başlamıştır. Metin madenciliği veri madenciliğinin bir alt disiplini olarak düşünülebilir. Hem veri madenciliği hem de metin madenciliği özünde bilgi yığının içinden anlamlı, değerli bilgiler çıkarmayı amaçlar. Blog'ların çoğalması, metin formatındaki bilgilerin üretim hızı metin madenciliği alanındaki çalışmaların yoğunlaşmasına neden olmuştur. Metin madenciliğinde kullanılan metinler farklı yapıda ve boyutlarda olabilmektedir. Bu nedenle metin madenciliği süreci zor bir iştir. En yaygın metin madenciliği uygulamaları spam e-posta tespiti, bir metinin özetinin çıkarılması veya bir metindeki duygunun belirlenmesi yani duygu analizinin yapılması şeklindedir. Bir metinde madencilik yapılacağı zaman metin diline göre doğal dil işleme disiplininden

faydalanmak gerekebilir. Günümüzde birçok metin madenciliği çalışmaları yapılmaktadır. Bu çalışmalara ek olarak ses madenciliği gibi yeni disiplinler de popüler olmaktadır.

Yıldız ve diğerleri çalışmalarında farklı alanlardan toplanan veri seti üzerinde yeni bir metin sınıflandırma yaklaşımı ile metinleri sınıflandırma işlemi gerçekleştirmişlerdir(Yıldız ve diğ. 2007). Hürriyet, Vatan ve Sabah gibi günlük gazetelerinin ekonomi, magazin, spor ve siyaset konularındaki köşe yazıları veri seti olarak toplanmıştır(toplamda 1150 doküman). Kelimelerin metinler içerisindeki ağırlıklarını kullanmak yerine, sınıflardaki(ekonomi, spor ve siyaset gibi.) ağırlıkları kullanılmıştır. Daha sonra metinde geçen bütün kelimelerin sınıf ağırlıkları toplanmıştır. Bu değerler normalize edilerek metin için yeni özellik vektörü oluşturulmuş. Oluşturulan bu özellik vektörüne göre metinler 5 farklı sınıfa ayrılmıştır. Kullanılan farklı algoritmalarla genel olarak %90 sınıflandırma başarımı elde edilmiştir.

İlhan ve diğerleri çalışmalarında metin madenciliği ile soru cevaplama sistemi gerçekleştirilmiştir(İlhan ve diğ. 2008). Doğal dil işleme alanı içerisinde yer alan biçimsel analiz yöntemiyle kullanıcı tarafından tedarik edilen soru metinleri işlenmekte ve soruya en yakın cevabın verilmesinin sağlanmasını amaçlamışlardır. Çalışma da sorular ve cevaplar için metin madenciliği yöntemleri uygulanmış(ön işleme, veri temizleme) daha sonra her soru ve cevaptaki kelimeler vektör uzay modeli kullanılarak ağırlıklandırılmış, kelimeler arasında benzerlikler belirlenmiştir. Sonuç olarak soruya uygulanan metin madenciliği yöntemlerinden sonra soru vektör haline getirilmiş ve veri tabanında bulunan vektörlerle karşılaştırılarak en yakın(benzerlik açısı hesaplanmış) cevap verilmiştir.

Dolgun ve diğerleri yapısal olan veri modeline, yapısal olmayan veri kümesinin yapısal hala getirildikten sonra modele eklenerek model başarımının analiz edilmesini amaçlamışlardır(Dolgun ve diğ. 2009). Çalışmalarında 2070 müşterinin bilgilerinden oluşan ve 17 değişkene sahip yapısal veri kümesine ait bir model oluşturulmuş. Daha sonra yapısal olmayan metin formatındaki müşteri bilgileri bazı yöntemlerle yapısal hale getirilmiş. Yapısal hale getirilen bu veriler diğer model verileriyle birleştirilmiş ve oluşturulan yeni modelin diğer model ile kıyaslaması yapılmıştır. Metin

madenciliği yardımıyla yapısal hale getirilen modelin daha başarılı olduğu kayıt edilmiştir.

Karadağ ve Takçı birçok kategoriden oluşan haber sitelerinin haberlerini, benzerliklerine göre tespit eden bir sistem geliştirmişlerdir(Karadağ ve Takçı 2010). Sistem metin madenciliği yardımıyla metinleri benzerliklerine göre sınıflandırmıştır. RSS(Really Simple Syndication) kaynaklarından alınan yaklaşık 370.000 sayısındaki metin üzerinde metin madenciliği önişlemleri gerçekleştirilmiştir. Bu ön işlemler metin içindeki HTML(Hypertext Markup Language) etiketlerinin temizlenmesi, noktalama işaretlerinin temizlenmesi vb. işlemlerdir. Daha sonra metindeki kelimeler için doğal dil işleme yöntemleri(kelime eklerinin silinmesi, köklerin belirlenmesi vb. işlemler) uygulanarak her haber metni için etiket listesi oluşturulmuştur. Etiket listesi oluşturulan haber metinlerin birçok farklı frekans yöntemiyle benzerlikleri belirlenmiştir.

Aşlıyan ve Günel metin sınıflandırmak için En Yakın Komşu ve K-En Yakın Komşu algoritmalarını kullanmışlardır(Aşlıyan ve Günel 2010). 5 farklı sınıf için 125 dokümandan oluşan derlem yardımıyla eğitim verisi oluşturulmuş. Her sınıf için öznitelik vektörleri oluşturularak veri tabanında saklanmıştır. Test için hazırlanan dokümanlar içinde öznitelik vektörleri oluşturulmuş ve En Yakın Komşu ve K-En Yakın Komşu metotlarıyla dokümanlar sınıflandırılmıştır. En Yakın Komşu metodu daha başarılı oluş ve tüm sınıflar için yaklaşık %88,4 oranında başarı kaydedilmiştir.

Ergün doktora tezinde internet üzerinden satış yapan bir e-ticaret sitesindeki ürün yorumlarının olumlu veya olumsuz olarak sınıflandırmıştır(Ergün 2012). Yorumlar üzerinde doğal dil işleme teknikleri uygulanmış, kelimeler morfolojik olarak çalışılmıştır. Bir ürüne ait bir özelliği niteleyen sıfatlar belirlenmiş ve kategorize edilmiştir. Geliştirilen yazılım yardımıyla yorumlardan otomatik olarak bilgi çıkarımı sağlamak için Türkçe dilinin dilbilgisi kuralları ve cümle yapısı referans alınarak ağaç yapısı oluşturulmuştur. Bu ağaç üzerinde geliştirilen yazılımda Derinlik Öncelikli Algoritma kullanılarak istenilen sonuca ulaşılmıştır.

Döven tez çalışmasında metin madenciliği yöntemini kullanarak birden fazla dokümanın bir biri arasında ki benzerliklerin bulunmasını tespit etmeyi amaçlamıştır(Döven 2013). Bu işlemleri gerçekleştirmek için geliştirmiş olduğu

uygulama dokümanlar arasındaki benzerlikleri Jaccard ve Cosinüs yöntemleriyle ifade etmiştir. Bu yöntemlerle n sayıdaki dokümanın bir birine olan benzerliklerini bulmaya çalışmıştır. Geliştirmiş olduğu masaüstü uygulamasıyla dokümanların bir birine olan benzerliklerini kelime tabanlı göstermiştir.

2.4 Hadoop ve Mahout

Hadoop bir büyük veri işleme teknolojisidir. Mahout ise Hadoop üzerinde çalışan ve makine öğrenmesi algoritmalarının Hadoop teknolojisi için yazılmış halini içeren bir framework'tür. Hadoop ve Mahout teknolojileri ilerleyen bölümlerde ayrıntılı bir şekilde ele alınacaktır. Hadoop çok büyük boyuttaki yapısal olmayan verileri işleyebilmekte ve güçlü bir hesaplama altyapısı sunmaktadır. Bunun yanında Hadoop teknolojisini kullanmak düşük donanım maliyetli bilgisayarlar yardımıyla sağlanabilmektedir. Literatürde Hadoop ile yapılan birçok çalışma mevcutken Mahout teknolojisiyle yapılan çalışmalar Hadoop'a nazaran daha az sayıdadır.

Hadoop dağıtık hesaplama ortamı sunmaktadır. Dağıtık hesaplama ortamı için de dağıtık dosya sistemini kullanmaktadır. O'Malley 2008 yılında yapmış olduğu çalışmayla Hadoop kümesi üzerinde terabayt büyüklüğündeki verileri sıralamak istemiştir(O'Malley 2008). Bunun için 3 farklı uygulama yazmış bu uygulamalardan birincisi dağıtık olarak veri üretmekte, ikincisi üretilen veriyi sıralamakta ve üçüncüsü de verinin sıralanıp sıralanmadığını kontrol etmektedir. Uygulama yardımıyla 10 milyon satır veri oluşturulmuştur. O'Malley bu çalışmayı 910 node'tan oluşan Yahoo kümesi üzerinde gerçekleştirmiştir. Hadoop bu devasa veriyi yalnızca 3.48 dakikada sıralamıştır.

Esteves ve diğ.(2011) çalışmalarında büyük veri niteliğindeki bir veri kümesi üzerinde K-Means kümeleme algoritmasını kullanarak Hadoop kümesinin performansını ve diğer istatistikleri incelemişlerdir. Veri kümesi olarak 4.940.000 tane kayıttan oluşan ve toplamda 1.1 GB büyüklüğündeki bir veri setini kullanmışlardır. Bu çalışmayı iki ayrı Hadoop kümesi üzerinde yapmışlar, birinci küme 5 node'a sahip ikinci küme ise bir node'a sahiptir. Deneylerinde veri setini boyutları farklı küçük alt bölümlere bölmüşler ve her deneyi her iki Hadoop kümesi üzerinde gerçekleştirmişler. Sonuç istatistiklerinden biri küçük boyuttaki bir veri kümesi üzerinde tek node'luk

kümenin performansı daha iyiyken, büyük boyutlu veri setinde 5 node'luk kümenin performansı daha iyidir. Bir diğer deney sonucu ise bir Hadoop kümesindeki node sayısının arttırılması her iş parçacığının çalışma süresini kısaltmasıdır.

Demir tez çalışmasında Hadoop'un kullanmış olduğu map-reduce programlama modeline göre görüntü işleme için yeni bir eklenti geliştirmiştir(Demir 2012). Hadoop dağıtık bilgisayarlar üzerinde map-reduce programlama tekniğini kullanmaktadır. Bu teknik az sayıda çok büyük boyutlu dosyalarda ciddi performans iyileştirmeleri sağlamaktadır. Geliştirilmiş olan eklentide bir yöntem olarak küçük boyutlu dosyaları birleştirerek daha büyük boyutlu dosyalar haline getirdikten sonra işlenmesini sağlamaktadır. Geliştirilen eklenti sanal olarak kurulan 6 node'tan oluşan dağıtık bir bilgisayar kümesi üzerinde çalıştırılmış ve test edilmiştir. İmgelerin birleştirilerek işlenmesinin daha iyi performans sağladığı kaydedilmiştir.

Sahu çalışmalarında K-means kümeleme algoritmasının kullanmış olduğu mesafe ölçüm tekniklerinden kosinüs uzaklık ölçüm yöntemi üzerinde bazı iyileştirmeler yapmışlardır(Sahu ve Mohan 2014). Algoritma için wikipedia makalelerinden oluşan 1.64 GB boyutundaki veri setini kullanmışlardır. Altyapı olarak Hadoop tercih edilmiş. Hadoop üzerinde makine öğrenme algoritmalarını map-reduce yöntemiyle çalıştıran Mahout kullanılmıştır. Hadoop kümesi 6 node'tan oluşmaktadır. Önerilen yeni uzaklık mesafesi ölçüm yönteminin daha başarılı sonuçlar verdiğini belirtmişlerdir.

Kayım tez çalışmasında Hadoop üzerinde K-Means ile DBSCAN(Density-based spatial clustering of applications with noise) algoritmasının paralelleştirmesinin veri analizinde performansını incelemiştir(Kayım 2015). Bunun yanında Hadoop ekosisteminin birer üyesi olan Impala ve Hive teknolojilerinin performansları test edilmiştir. Impala ve Hive, Hadoop teknolojisinde, büyük veri üzerinde SQL(Structured Query Language) sorgularının çalıştırılmasına altyapı sağlayan teknolojilerdir. K-means algoritmasının paralelleştirmesinin sonucu yaklaşık olarak, paralel olarak çalıştırılan bilgisayarların âdeti kadar performans iyileştirme gözlemlenmiştir. Bunun haricinde Hadoop sistemindeki diğer teknolojiler işlenmiştir.

Aydın ve Hallaç (2015) yapmış oldukları çalışmalarında Hadoop üzerinde makine öğrenmesi yöntemiyle doküman sınıflandırmışlardır. Toplamış oldukları

dokümanları spor, ekonomi, kültür vb. sınıflara otomatik bir şekilde ayırmışlardır. Doküman sayısının sınıflandırma süresi performansına etkisi ve doküman sayısının doğru sınıflandırma performansı üzerindeki etkisi araştırılmış ve sonuçlar grafiksel olarak ifade edilmiştir. Testler için tek node'luk Hadoop ve kümeden(4 bilgisayar) oluşan Hadoop altyapıları kullanılmıştır.

3. BÜYÜK VERİ

Bilgisayar teknolojisinin çok hızlı ilerlemesi ve hemen hemen her şeyin dijital olarak kayıt altına alınması yeni kavramların ortaya çıkmasına neden olmuştur. Bu kavramlardan biri de "Big Data"(Büyük Veri) olmuştur. Büyük veri kavramının oluşmasındaki en önemli faktör internet ağı olmuştur. İnternet ile birlikte hayatımızdaki bir çok şeyi online (çevrimiçi) yapıyoruz. Hastane işlemleri, alışveriş, yemek siparişi, bilet(uçak, otobüs) işlemleri, banka işlemleri vb. birçok kişisel işlerimizi internet üzerinden yapmaya çalışıyoruz. Kişisel hayatımızdaki her anı sosyal medya üzerinden paylaşmak ve kayıt altına almak istiyoruz. Kayıt altına almak istediğimiz veriler veya internet üzerinden yaptığımız işlere ait veriler değişik formatlarda ve yapılarda olabilmektedir. Örneğin; fotoğraf, video, ses dosyası veya bir metin olmanın yanında ayrıca herhangi bir sensördeki sayısal rakamlardan oluşan bir dizi veri de olabilmektedir. Bu değişik formattaki verileri depolamak, analiz etmek ve yönetmek için günümüzde klasik veri tabanı yönetim sistemleri yetersiz kalmaktadır.

Büyük veri kapsam olarak sosyal medya verileri, web sitelerine ait log bilgileri, e-postaları, sensörlerin ürettiği veya kameraların ürettiği verileri içermektedir. Büyük verinin içeriğini ve büyüklüğünü ifade eden istatistiklerden biri olarak 2016 yılında bir dakika içerisinde dünyada;

- WhatsApp üzerinden 20.8 milyon mesaj gönderildi.
- 701.389 kişi Facebook'ta oturum açtı.
- 150 milyon e-posta gönderildi.
- Snapchat üzerinden 527.760 fotoğraf paylaşıldı.
- Google üzerinden 2.4 milyon arama yapıldı.
- Instagram üzerinden 38.194 fotoğraf paylaşıldı.
- AppStore üzerinden 51.000 indirme işlemi yapıldı.
- Youtube'ta 2.78 milyon video izlendi.
- Amazon 203,596 $ satış yaptı(Mitchell 2016).

3.1 Büyük Veri Tanımı

Büyük veri kavramına dair literatürde birçok tanım mevcuttur. 2001 yılından önceki yıllarda megabayt ve gigabayt büyüklüğündeki veriler büyük veri olarak ifade edilmekteydi. 2001 yılında Laney, D. ve arkadaşları yayınlamış oldukları çalışmalarında dijital olarak üretilen büyük boyuttaki verilere dair 3 temel özelliği ele almışlardır(Chen ve diğ. 2014). Bu özellikler; verinin hacmi, çeşitliliği ve üretim hızı olmuştur. Literatürde bu özellikler 3V (Volume, Velocity ve Variety) modeli olarak ifade edilmektedir. Bu özellikleri taşıyan veri kümelerini büyük veri olarak tanımlamışlardır. Veri kümelerinin 3V modeli haricinde hem yapısal hem de yapısal olmayabileceklerini ifade etmişlerdir. IBM ve Microsoft gibi büyük kuruluşlar 3V modelinin önümüzdeki 10 yıl boyunca büyük veri kavramını tanımlayacağını öngörmüşlerdir(Chen ve diğ. 2014).

2010 yılında Apache Hadoop, normal bilgisayarda depolanamayan, yönetilemeyen ve işlenemeyen verileri büyük veri olarak tanımlamıştır(Chen ve diğ. 2014). Normal bilgisayar günlük hayatta kullanılan kişisel bilgisayarlardır. Bu bilgisayarlarda çok yüksek işlem yapma kapasiteleri ve depolama yetenekleri yoktur. Bu nedenle bu bilgisayarlar üzerinde kullanacağımız veri tabanı yönetim sistemleri büyük veri olarak nitelendirdiğimiz veri kümelerini işleme konusunda istenilen performansı sağlayamayacaklardır. Büyük veri olarak nitelendirilen veri kümeleri genellikle terabayt ve petabayt boyutundaki veri kümeleridir(Manyika ve diğ. 2011).

2011 yılında IDC(International Data Corporation) büyük veri teknolojilerini, büyük veriden değerli/anlamlı bilgiler çıkaran teknolojiler olarak ifade etmiştir(Gantz ve Reinsel 2011). Bu tanımla daha önce büyük veri için kullanılan 3V modeline bir V(Value) daha eklenmiş bulunmaktadır. IDC büyük veri kümesinin içinden anlamlı bilgiler çıkarmanın büyük veri tanımının bir parçası olduğunu ifade etmiştir.

Wikipedia büyük veri kavramı hakkında, normal bilgisayarlardaki yazılımlarla yönetilemeyen, depolanamayan ve analiz edilemeyen büyük boyuttaki veri kümeleri olarak tanımlamıştır (Wikipedia Büyük Veri 2016). Özetlemek gerekirse büyük veri; içerisinde yapısal, yapısal olmayan veya yarı-yapısal verileri içeren, çok yüksek büyüme hızına sahip, normal bilgisayarlar tarafından depolanamayan, yönetilemeyen, analiz edilemeyen, üzerinde veri madenciliği yapılamayan veri kümeleri olarak

tanımlanabilir. Normal bilgisayarlar haricinde yüksek kapasiteli hesaplama yapmaya olanak sağlayan süper bilgisayarlar, donanım maliyetleri açısından herkes tarafından kullanılamayan bilgisayarlardır. Bu nedenle düşük donanım maliyetiyle büyük veri işlemeye olanak sağlayacak teknolojilere ihtiyaç olmuştur. Günümüzde bu ihtiyaca çözüm olabilecek teknolojiler mevcuttur.

3.2 Büyük Veri Özellikleri

Büyük veri tanımlarıyla birlikte gelen bazı büyük veri özellikleri aşağıdaki gibidir.

- Çeşitlilik(Variety): Gelişen teknolojinin sonucu olarak günümüzde birçok dijital veri üreten cihaz bulunmaktadır. Günlük hayatta kullanılan güvenlik kameralarının görüntülerinden deprem tespit enstitülerinde bulunan sinyallerin üretmiş oldukları verilere kadar birçok farklı yapıdaki veri anlık olarak üretilmektedir. Bu cihazların üretmiş oldukları verilerin yapısı/tipi farklı olmaktadır. Verilerin yapısı; yapısal veri, yapısal olmayan veri veya yarı-yapısal veri olabilmektedir. Bu veriler üzerinde işlem yapabilmek için bazen çok uzun veri dönüştürme işlemleri kaçınılmaz olmaktadır. Büyük veri kavramı bu yapıları farklı olan tüm verileri kapsamaktadır.
- Veri Büyüklüğü(Volume): Büyük veri denilince aklımıza gelen ilk büyük veri özelliğidir. Dünyada günlük olarak yaklaşık 2.3 trilyon veri üretilmektedir. Büyük veri kavramı için veri büyüklüğü denilince gigabayt düzeyindeki veri tabanlarından veya veri ambarlarından bahsedilmemektedir. Veri büyüklüğü olarak genellikle terabayt veya petabayt büyüklüğündeki veri kümesleri kastedilmektedir.
- Hız(Velocity): Büyük verinin diğer özelliği, verinin üretim hızıdır. 2015 yılında bir dakika içerisinde Twitter üzerinden yaklaşık 422340 tane tweet atılmıştır(Allen 2016). İnternet üzerindeki cihazlar arasında saniyede milyonlarda veri akışı sağlanmaktadır. Gelişen teknolojik alt yapı üretilen bu veri akışının daha hızlı iletilmesi için çeşitli protokoller sunmaktadır. Bu şekilde verilerin üretim hızı daha da arttırılmaktadır.

- Verinin Değeri(Value): Son yıllarda büyük veri kavramı için en çok ilgi gören büyük veri özelliğidir. Bu kadar hızlı artan ve büyüklüğü bu kadar devasa olan veri kümelerinin içinden anlamlı ve değerli bilgilerin nasıl çıkarılacağı sorusunun cevabını bulmayı amaçlar. Bu sorunun cevabı oluşturmak için birçok yazılım ürünü geliştirilmektedir. Bu ürünlerden biri de Hadoop çatısı üzerinde çalışan Mahout teknolojisidir. Hadoop ve Mahout teknolojileri ilerleyen bölümlerde detaylıca açıklanacaktır.

Büyük veri kavramının içerisini dolduran, kavramın çerçevesini oluşturan 4V modelidir. Şekil 3.1'de büyük verinin özellikleri ile olan bağlantılarını gösteren görsel bulunmaktadır. Dört özelliğin birleşmesiyle büyük veri kavramı oluşmaktadır.

Şekil 3.1: Büyük veri özellikleri

4. BÜYÜK VERİ ARAÇLARI

Büyük verinin hızlı büyümesi, üretilen verinin özellikleri bu verilerin işlenmesinde yeni teknolojilere ihtiyacı doğurmuştur. Büyük veri kavramının özellikleri göz önünde bulundurularak bu ihtiyaca cevap olacak teknolojiler üretilmiştir. Bu teknolojiler genellikle çok büyük veri kümelerine sahip özel sektör firmaları tarafından geliştirilmişlerdir. Bu teknolojilerden bazıları, Apache Hadoop, Apache Spark, Apache Mahout, Apache Hive, Apache Nutch, Apache Pig şeklindedir. Apache kelimesiyle başlayan bu teknolojiler ayrı ayrı Apache Software Foundation(ASF)(Apache Yazılım Kurumu) tarafından desteklenen ve geliştirilen projelerdir.

4.1 Apache Hadoop

Hadoop verileri dağıtık ve paralel olarak işlemek amacıyla ASF tarafından geliştirilen bir projedir(Shafer ve diğ. 2010). Hadoop büyük veriyi en verimli şekilde işlemeyi amaçlayan bir proje olarak geliştirilmeye başlanmıştır. Java programlama diliyle açık kaynak kodlu olarak geliştirilmiştir. Hadoop projesinin iki önemli bileşeni vardır. Bu bileşenler Hadoop Distributed File System(HDFS) dosya sistemi ve map-reduce programlama tekniğidir. HDFS Hadoop için özelleştirilmiş bir dosya sistemidir. Map-reduce ise dağıtık sistemler üzerinde yüksek başarım sağlayan bir paralel programlama yaklaşımıdır. Hadoop bu bileşenlerden oluşan bir framework'tür. Bu framework bir bilgisayar veya binlerce bilgisayardan oluşan bir küme üzerinde de çalıştırılabilir. Bu şekilde yüzlerce bilgisayar üzerinde dağıtık olarak çalıştırılarak, yüksek boyutlu verileri işlemeye olanak sağlamaktadır.

Hadoop düşük özellikli bilgisayarlar üzerinde de çalıştırıldığında yüksek performans sağlayabilmektedir. Bunun yanında Hadoop kümesinin ölçeklenebilir olması, yani istenildiği zaman Hadoop kümesine kolayca yeni bir bilgisayar eklenebildiği için Hadoop büyük veri araçları arasından tercih edilen çözüm olmaktadır. Hadoop'un sağlamış olduğu avantajlardan bir diğeri de veri türü bağımsızlığıdır. Hadoop üzerinde birçok veri türü saklanabilir ve işlenebilir.

Hadoop günümüzde hem ticari amaçlar için hem de akademik çalışmalar için çokça kullanılmaktadır. Özel sektör de ilk olarak 2006 yılı Mayıs ayında Yahoo 300 bilgisayardan oluşan bir Hadoop kümesi oluşturmuştur. Çok hızlı üretilen veriyi daha iyi işleyip depolamak için Yahoo 2009 yılında 17 farklı küme ve toplamda 24.000 bilgisayardan oluşan bilgisayar altyapı sisteminde Hadoop kullanmıştır (Wikipedia Hadoop 2016). Benzer şekilde birçok büyük firma Hadoop teknolojisinden faydalanmaktadır. Bu firmaların başında EBay, Facebook ve IBM gelmektedir(Kayım 2015).

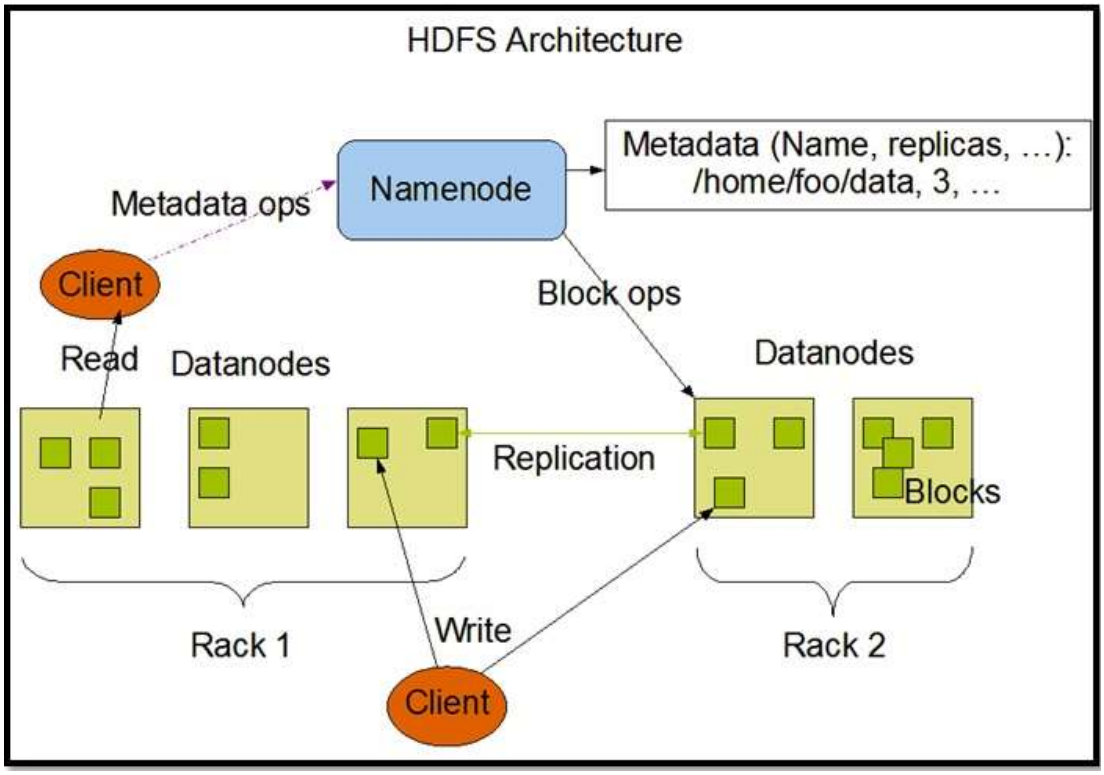
Hadoop yapısı gereği çok büyük dosyaları işlemek saklamak için geliştirilmiştir. Hadoop ile petabayt düzeyindeki veriler depolanıp işlenebilir. Hadoop üzerinde depolanan veriler daha küçük boyutlara bölünerek Hadoop kümesini oluşturan bilgisayarlarda saklanır. Verilerin parçalanarak kümedeki bilgisayarlar üzerinde saklanması, Hadoop'un veri yerelliği özelliğiyle ilgilidir. Bu verilerin saklanması ve işlenmesi iki kontrol mekanizmasıyla yapılmaktadır. Bu mekanizmalar NameNode ve DataNode yapısıdır.

4.1.1 Hadoop Distributed File System(HDFS)

2003 yılından önce Google firması kullandıkları veri yönetim teknolojilerinin yetersiz kalması nedeniyle yeni yatırımlar yapmıştır. Bu yatırımlardan biri de dosya sistemiyle ilgili olmuştur. 2003 yılında Google kendi dosyalarını saklamak ve sorgulamak için kendisine özgü dosya sistemini oluşturmuştur. Bu dosya sistemi Google File System(GFS)(Google Dosya Sistemi) olarak 2003 yılında tanıtılmıştır(Ghemawat ve diğ. 2003). GFS büyük boyuttaki dosyaları dağıtık olarak küme üzerinde saklamayı ve verinin paralel olarak işlenmesini sağlar. GFS ile verilerin farklı bilgisayarlarda yedeklenmesi sağlanmıştır. Bunun yanında veriler 64 megabayt boyutunda alt bloklara bölünmüş olarak bilgisayarlarda saklanmıştır(Ghemawat ve diğ. 2003). GFS'in çalışma performansından etkilenen Doug Cutting, map-reduce tekniğinden de faydalanarak 2008 yılında Hadoop Distributed File System(HDFS)(Hadoop Dağıtık Dosya Sistemi)'ini geliştirdi(Demir 2012). HDFS map-reduce görevleri için tasarlanmış bir yapıdır. HDFS uygulama verilerini ve dosya

sistemi metadatalarını farklı yerlerde saklamaktadır. HDFS çok büyük boyutlu dosyaların saklanmasına olanak vermektedir.

HDFS Hadoop sisteminin en önemli parçasıdır. Hadoop küme üzerinde çalıştırılmak istenildiğinde mimari olarak master/slave mimarisini kullanır. Master/slave mimarisinde küme içindeki bir bilgisayar master(yönetici) olur diğer bilgisayarlar slave(işçi) node'lar olarak ifade edilir. Master genel küme yönetiminden ve görevlerin/işlerin slave node'lara dağıtılmasından ve sonuçların birleştirilmesinden sorumludur. Slave node'lar ise master tarafından kendisine verilen görevleri yapmakla sorumludur. HDFS bu yapı üzerinde çalışmaktadır ve iki önemli bileşeni vardır. Bunlar NameNode ve DataNode'dır. Hadoop kümesinde en fazla bir tane NameNode bulunabilir, diğer node'lar DataNode olarak işlev görürler(Wikipedia Hadoop 2016). Şekil 4.1'de genel HDFS yapısı ve çalışma şekli gösterilmiştir. Kullanıcı NameNode'a yapmak istediği işlem doğrultusunda istekte bulunur, NameNode istekle ilgili verilerin metadata bilgilerini ve tanımladığı görevi DataNode'a gönderir. DataNode işin/görevin sonucunu NameNode'a aktarır.



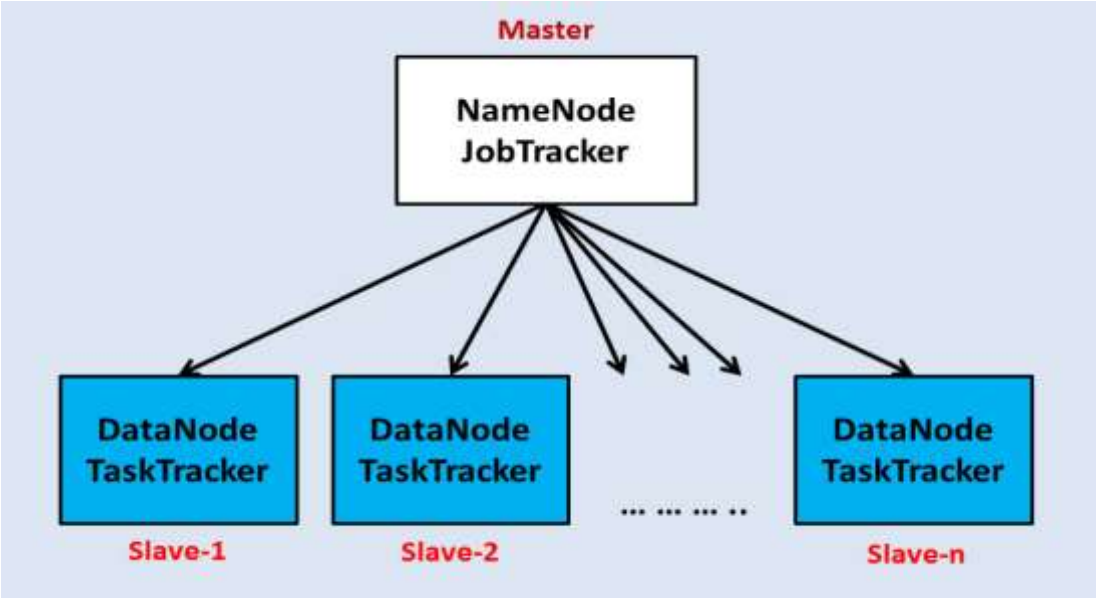
Şekil 4.1: HDFS mimarisi (HDFS Design, 2016)

4.1.1.1 NameNode

Hadoop ile petabayt düzeyindeki veriler depolanıp işlenebilir. Hadoop üzerinde depolanan veriler daha küçük boyutlara bölünerek hadoop kümesini oluşturan bilgisayarlarda saklanır. Dosyalar Hadoop tarafından varsayılan olarak 64 megabayt'lık küçük alt parçalara bölünür, oluşan bu alt parçalara blok denir(Shafer ve diğ. 2010). NameNode oluşturulan bu bloklarla ilgili işlemlerden sorumludur. Bu blokların oluşturulması, bu blokların slave node'lar üzerindeki dağılımlarından ve dosya erişimlerinden sorumludur. NameNode ayrıca dosya sistemindeki dosyaların açılması kapatılması veya yeniden isimlendirilmesi işlemlerini yürütür. NameNode master node üzerinde çalışır ve slave node'larda saklanan dosyaların metadatalarını saklar. Bu metadatalar her dosyaya ait blokların saklandığı adresler vb. bilgilerden oluşmaktadır.

Hadoop'un çekirdek yapısı NameNode node'unun düzenli olarak yedeğinin alınmasını sağlamaktadır. Alınan bu yedeğe Secondary NameNode denilmektedir. Eğer Hadoop kümesinde bir nedenden dolayı NameNode çalışması sonlanırsa, yerine görevini sürdürecek olan Secondary NameNode çalışmaya başlayacaktır.

Hadoop kümesinde master node üzerinde çalışan NameNode'un yanında bir de JobTracker(görev takipçisi) vardır. JobTracker Hadoop kümesindeki görevlerin yürütülmesinden sorumludur. Slave node'lara yeni iş verilmesi, verilen işin durumunu kontrol edilmesi gibi işlemleri takip etmektedir(JobTracker 2016). Ayrıca JobTracker verinin map ve reduce fonksiyonları için parçalara ayırılmasından sorumludur. Şekil 4.2'de JobTracker ile TaskTracker arasındaki çalışma ilişkisi gösterilmiştir. TaskTracker DataNode'ları üzerinde çalışmaktadır. Örneğin; bir metindeki "a" harflerinin sayısı bulunmak istendiğinde, JobTracker yeni bir görev oluşturur. Bu görevi DataNode'lara bildirir, görevin yanında her node'da hangi veri bloğunun bulunduğu bilgilerini gönderir. DataNode üzerinde çalışan TaskTracker gelen görevi karşılar, gerekli hesaplamaları yapar. Hesaplamalar sonucunda elde ettiği sonucu geri döndürür. Bu işlemler map-reduce yönteminin gerçekleştirimidir.



Şekil 4.2: JobTracker ve TaskTracker

4.1.1.2 DataNode

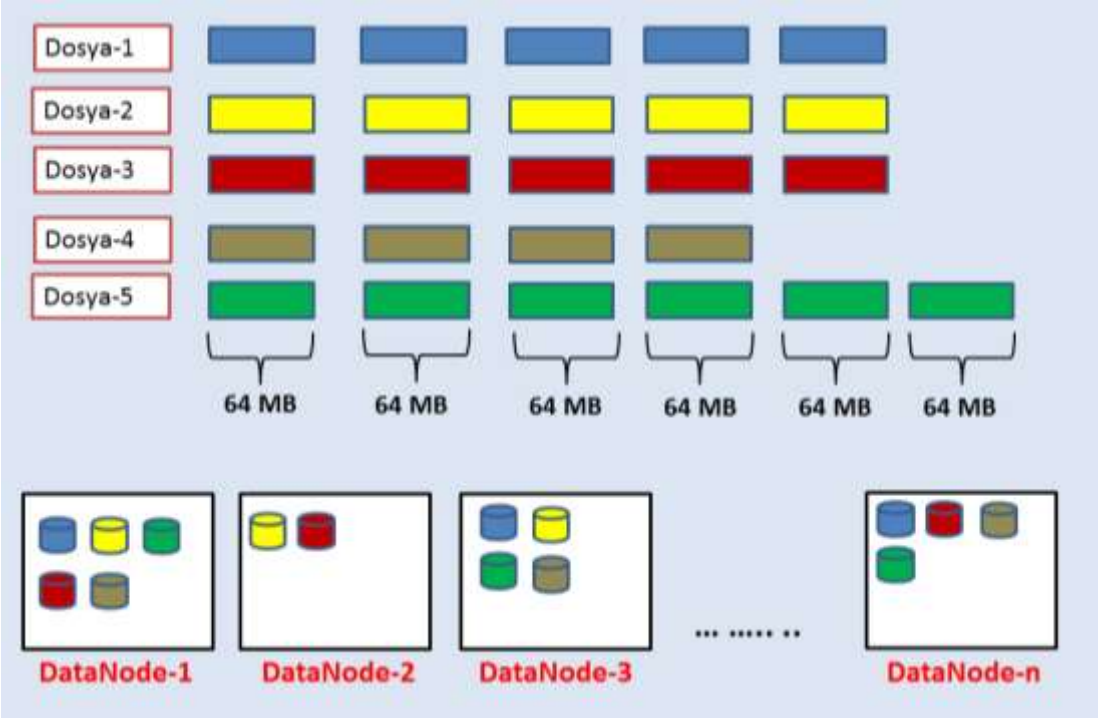
DataNode, Hadoop kümesinde verilerin saklandığı slave node'lar üzerinde çalışır. Görevi bulunduğu slave node üzerinde depolanan verilerin yönetimidir. Her DataNode veri bloklarının depolanması için gerekli servisleri HDFS dosya sistemine sağlar. NameNode depolanan bloklara ait bilgileri bu servisler yardımıyla DataNode'lardan almaktadır(Shvachko ve diğ. 2010). Her slave node için DataNode üzerinde saklanan veri bloklarının kopyaları diğer slave node'larda da saklanmaktadır. Bu şekilde verinin yedeklenmesini sağlamaktadır. Her bloğa ait kaç yedeğin alınacağı Hadoop dosyalarının içinden hdfs-site.xml dosyasında parametre olarak verilebilmektedir.

DataNode üzerinde bulunan TaskTracker, JobTracker tarafından gönderilen işlerin DataNode üzerinde çalıştırılmasından sorumludur. Bu işler map veya reduce operasyon işlemleridir. Şekil 4.2'de TaskTracker'ın ve JobTracker'ın çalışma şekilleri gösterilmiştir.

Şekil 4.3'te verinin DataNode'lar üzerinde nasıl saklandığı gösterilmiştir. İlk işlem olarak 5 ayrı dosya öncelikle 64 megabaytlık alt bloklara bölünür. Her blok bir DataNode tarafından depolanır, ayrıca her bloğun kopyası farklı DataNode'lar

üzerinde saklanır. Burada her DataNode farklı birer slave sunucusunda bulunmaktadır. Bir blok gereğinden fazla yedeklenmişse NameNode tarafından tespit edilir ve DataNode'a bildirilir, DataNode saklamış olduğu fazla yedeği siler(Shvachko ve diğ. 2010).

Hadoop'un sağladığı avantajlarından biri de sonradan kümeye yeni bir bilgisayar(slave) eklenebilmesidir. Sonradan eklenen bilgisayar üzerinde saklanan veri bulunmayacağından hadoop kümesindeki DataNode'larda saklanan verilerin boyutları dengeli olmayacaktır. Bu dengeyi sağlamak için HDFS balancer(dengeleyici) aracını kullanmaktadır(Shafer ve diğ. 2010). Bu araç hangi node'ta depolanan dosya fazla ise eksik olan node'lara taşımaktadır. Bu durumda DataNode'lar üzerinde saklanan veriler dengelenmiş olmaktadır. Şekil 4.3'teki DataNode-2 sonradan eklenen bir slave olduğundan üzerinde saklanan veri diğer node'lardan daha azdır. Balancer fonksiyonu bu durumu fark ettiğinde, diğer node'lardan hangisinde saklanan veri bloğu fazla ise ilgili bloktan yeni eklenen slave bilgisayara dosyaları taşımaktadır. Bu süreçteki blok bilgilerini ve slave node bilgileri NameNode'a aktarmaktadır. Bu şekilde NameNode üzerinde saklanan metadata bilgileri güncel kalmaktadır.



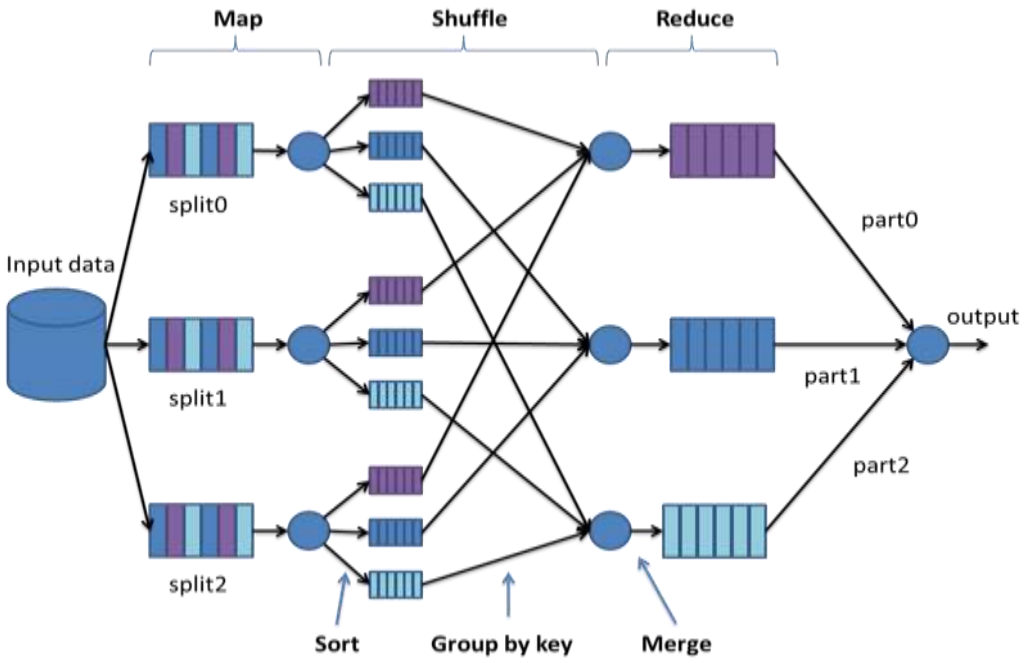
Şekil 4.3: DataNode üzerinde verilerin saklanması

4.1.2 Map-Reduce

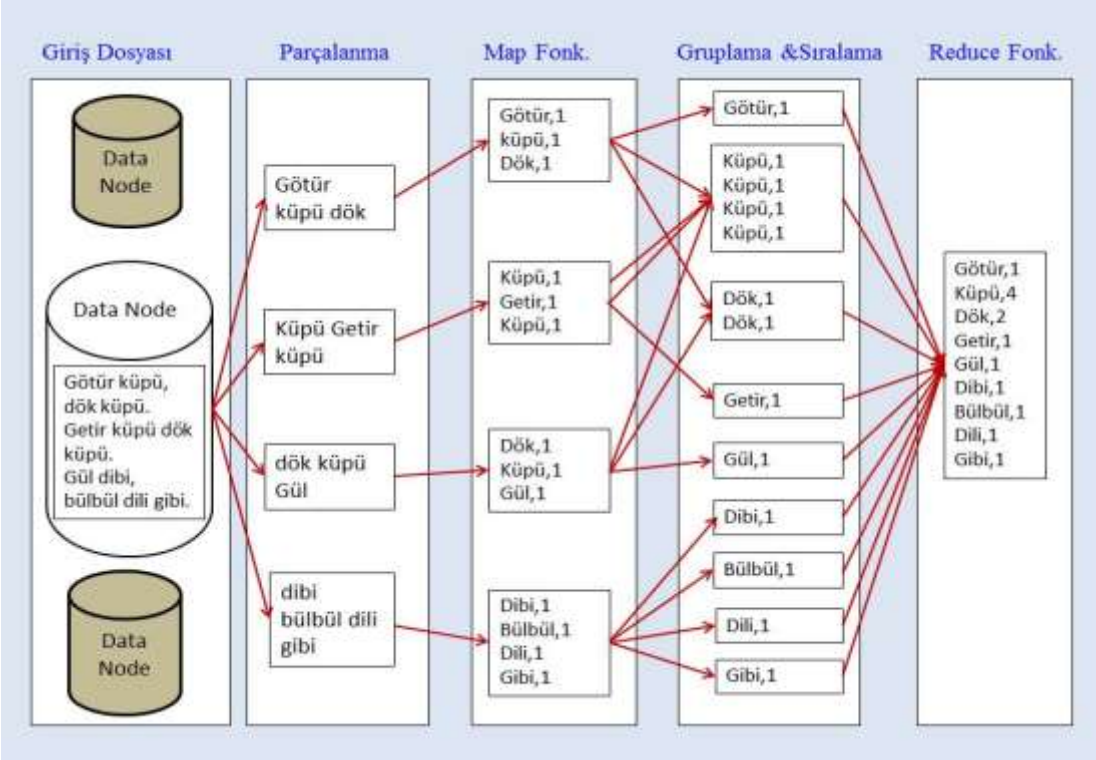
Map-reduce büyük veri kümelerini dağıtık bir mimari üzerinde verimli bir şekilde işlemeye olanak sağlayan bir framework'üdür. Bir başka deyişle map-reduce bir programlama modelidir. Map-reduce yaklaşımında işler map ve reduce kısımlarından(fonksiyonlarından) oluşur. Map fonksiyonu key/value(ahtar/değer) ikililerini parametre olarak alır ve yine key/value(ahtar/değer) değer çiftlerini üretir. Üretilen bu anahtar değer çiftleri reduce fonksiyonu için giriş parametreleridir(Shafer ve diğ. 2010). Reduce fonksiyonu parametre olarak almış olduğu aynı anahtar ikililerini birleştirerek sonuç üretir. Map-reduce işlerindeki hataları kontrol eden ve başarısız olan işleri tekrar çalıştırma gibi bir mekanizması vardır. Bu mekanizma yardımıyla her kullanıcı tarafından paralel programlama altyapısı kolaylıkla kullanılabilmektedir.

Map-reduce modelinin çalışma şekli şöyledir; büyük boyuttaki doküman öncelikle küçük boyutlardaki alt dokümanlara bölünmüş olarak HDFS üzerinde saklanmaktaydı, bu bloklar map fonksiyonun giriş parametreleridir. Her bir map fonksiyonu için giriş bloklarının sayıları yapılandırma ayarlarından Hadoop kullanıcı tarafından belirlenebilir. Her blok verisi map fonksiyonun giriş parametresi formatına dönüştürülür. Bu format key/value(anahtar/değer) ikilileridir. Bu anahtar değer ikileri map fonksiyonu tarafından sıralanır ve gruplanır. Sıralanmadan sonra oluşan bu veriye Intermadiate Output(orta çıktı) denir. Bu orta çıktı belleğe kaydedilir. Kayıt edilen bu veri reduce fonksiyonu için giriş parametreleridir. Reduce fonksiyonu gruplanmış ve sıralanmış olan bu orta çıktıyı parametre olarak alır ve indirgeme işleminden sonra yeni anahtar/değer ikilileri üretir. Bu işlemden sonra orta çıktı hafızadan silinir. Reduce fonksiyonun çıktısı HDFS dosya sistemine yazılır ve saklanır.

Şekil 4.4'te map-reduce modelinin çalışması gösterilmiştir. Bloklar halinde HDFS üzerinde saklanan veri alt bölümleri(split0,split1 vb.) map fonksiyonuna giriş değerlerini oluşturmaktadır. Map fonksiyonunun çıktıları sıralanıp gruplandıktan sonra recude fonksiyonuna parametre olarak gönderilmektedir. Reduce fonksiyonu almış olduğu anahtar değer ikililerini birleştirerek işlemi tamamlamaktadır.



Şekil 4.4: Map-reduce çalışma şekli (Map-reduce, 2016)



Şekil 4.5: Map-reduce uygulama örneği

Şekil 4.5'te bir map-reduce örneğinin görseli mevcuttur. Bu örnekte HDFS üzerinde depolanan bir metin dosyasındaki kelimelerin sayısının bulunması amaçlanmıştır. Parçalama aşamasında doküman alt parçalara ayrılmış ve map

fonksiyonuna parametre olarak gönderilmiştir. Map fonksiyonu her bir blok için kelime sayılarını hesapladıktan sonra orta çıktı üretir. Üretilen orta çıktı key/value formatındadır. Bu orta çıktı üzerinde gruplama ve sıralama işlemi uygulandıktan sonra yeni çıktı oluşturulur. Oluşturulan bu çıktı reduce fonksiyonu için giriş parametrelerini oluşturur. Reduce fonksiyonu map fonksiyonunun çıktılarını birleştirerek oluşan çıktı dosyasını HDFS dosya sistemine yazar ve map-reduce görevi sonuçlanmış olur. Map-reduce işinden sonra sonuç olarak "götür" kelimesinden 4 adet, "küpü" kelimesinden 4 adet, "dök" kelimesinden 2 adet, "getir", "gül", "dibi", "bülbül", "dili" ve "gibi" kelimelerinden birer adet olduğu görülmüştür.

Map-reduce modelinde oluşabilecek hataların toleransı hadoop tarafından sağlanmıştır. Kullanıcı yalnızca map ve reduce fonksiyonlarını yazarak dağıtık bilgisayarlar üzerinde hesaplamalarını gerçekleştirebilmektedirler. Bu da hadoop sisteminin çok verimli kullanılmasına ve tercih edilmesine neden olmuştur.

4.1.3 Hadoop Ekosistemi

Büyük veriyi yüksek performansla paralel işlemek için hadoop projesinin yanında birçok proje geliştirilmiştir. Bu projeleri hadoop projesini destekleyici projeler olarak düşünebiliriz. Bu projelerin başında; Apache Hive, Apache Nutch, Apahce Pig ve Apache Mahout gelmektedir. Bu projelerden Apache Mahout bu tez çalışması kapsamında kullanılmıştır.

4.1.3.1 Apache Mahout

Veriler üzerinde işlemler gerçekleştirip verilerden anlamlı bilgiler çıkarmak geçmişten günümüzde çok sık çalışılan bir konu olmuştur. Veri madenciliği veya alt bilimi olan metin madenciliği, web madenciliği, ses madenciliği bu süreci gerçekleştiren bilim alanlarıdır. Günümüzde madencilik için birçok uygulama ve algoritma geliştirilmiştir. Kullanılan uygulamaların yetenekleri çok büyük kapasiteli verileri işlemek için yetersiz kalmaktadır. Örneğin petabayt boyutundaki log bilgilerini bu teknolojilerle sınıflandırmak çok zaman alacaktır. Büyük verinin çözümlerinden biri olan Hadoop yardımıyla veri madenciliği yapmak için Hadoop geliştiricileri

tarafından Mahout projesi geliştirilmiştir. Mahout, veri madenciliği algoritmalarının büyük veri üzerinde map-reduce modeline göre çalışacak şekilde gerçekleştirimidir. Yani veri madenciliği algoritmaların map-reduce formatında yazılı bulunduran bir framework'tür.

Mahout, Hadoop üzerinde çalışan açık kaynak kodlu makine öğrenmesi framework'üdür. Mahout paralel veri madenciliği yapmaya olanak sağlamaktadır. Mahout temel veri madenciliği yöntemlerini gerçekleştirir. Bu yöntemler sınıflandırma, kümeleme ve öneri yöntemleridir(Hammond ve Varde 2013). Mahout için geliştirilen bir ara yüz bulunmamaktadır, komutlar terminal ekranına satır satır yazılır. Hadoop gibi Mahout projesi de java diliyle yazılmıştır. Mahout için herhangi bir makine öğrenmesi algoritması map-reduce modeliyle yazılıp geliştirilebilir.

Bu tez çalışması kapsamında Mahout makine öğrenmesi algoritmalarından Naive Bayes algoritması kullanılmıştır. Naive bayes istatistik tabanlı bir sınıflandırma algoritmasıdır.

4.1.3.2 Diğer Hadoop Destekleyici Çözümler

Hadoop projesinden sonra, Hadoop'un yetersiz kaldığı noktaları tamamlamak amacıyla birçok proje başlatılmıştır. Bu projeler bazen özel bir alandaki problemleri çözümlemek bazen de bir problem için daha iyi çözüm üretmek için geliştirilmişlerdir. Bu projelerden Hadoop'un destekleyici olanlarından bazıları aşağıda özetlenmiştir.

- Apache Hive: SQL kullanarak büyük boyutlu veriler üzerinde okuma yazma gibi işlemleri gerçekleştiren veri ambarı yazılımıdır(Hive 2016). Hive hadoop üzerinde map-reduce modeliyle yazılmış SQL komutlarının çalıştırılmasına imkân sağlamaktadır.
- Apache HCatalog: Hive projesinin önemli bir parçasıdır. HCatalog tablosu HDFS üzerinde depolanan kullanıcı verilerinin ilişkisel gösterimini sağlar. Saklanan verilerin formatı vb. bilgileri verir(Hadoop Ecosystem 2016).

- Apache Tajo: Tajo güçlü bir veri ambarı yazılımıdır. İlişkisel ve dağıtık bir veri sistemidir. Tajo HDFS üzerinde saklanan veriler üzerinde çok hızlı SQL sorguları yazmak için geliştirilmiştir(Tajo 2016).
- Apahce Pig: Hadoop üzerinde çalışan, HDFS altyapısını, map-reduce modelini kullanan, veri işlemeye olanak sağlayan Hadoop destekleyici bir projedir. Pig, Pig Latin dilini kullanarak veri üzerinde sıralama, birleştirme, sıralama, sorgulama gibi temel veri işleme operasyonların gerçekleştirilmesine olanak sağlamaktadır. Pig latin dili yüksek seviyeli bir dildir. Pig normal programlama dillerinde if ve döngü komutlarını içermez, pig veri akışına ve veri işlemlerine odaklı geliştirilmiştir(Pig 2016).
- Apache Nutch: Nutch, Apache Lucene için geliştirilmiş esnek ve ölçeklenebilir açık kaynak kodlu web tarayıcısı projesidir(Hadoop Ecosystem 2016).
- Apache HBase: Hadoop için geliştirilmiş olan ilişkisel olmayan büyük veri veritabanıdır. Java programlama diliyle yazılmıştır. HBase'in geliştirilmesinde Google tarafından geliştirilmiş olan BigTable projesinden esinlenilmiştir. HBase gerçek zamanlı olarak HDFS üzerinde veri yazma ve okuma işlemleri yapabilmektedir. HBase ilişkisel veri tabanlarında kullanılan SQL diline destek vermemektedir. 2010 Kasım ayında Facebook mesaj servisleri için kullanmaya başlamıştır(Hbase 2016).
- Apache Kudu: HBase gibi Hadoop için geliştirilen açık kaynak kodlu, dağıtık, kolon bazlı, ilişkisel verileri saklamaya yarayan bir projedir. Kudu projesinin temel amacı çok hızlı değişen veriler üzerinde çok hızlı analizler yapılmasına olanak sağlamaktır(Hadoop Ecosystem 2016).
- Apache Flume: Dağıtık, güvenilir sürekli akan veri işleme servisidir. Verileri verimli bir şekilde birleştirme, toplama ve taşınmaya olanak sağlar(Flume 2016).
- Apache Impala: Hive benzer ölçeklenebilir paralel veritabanı sistemidir. Hive göre daha hızlı sonuç getirmek gibi üstün yetenekleri vardır(Hadoop Ecosystem 2016).

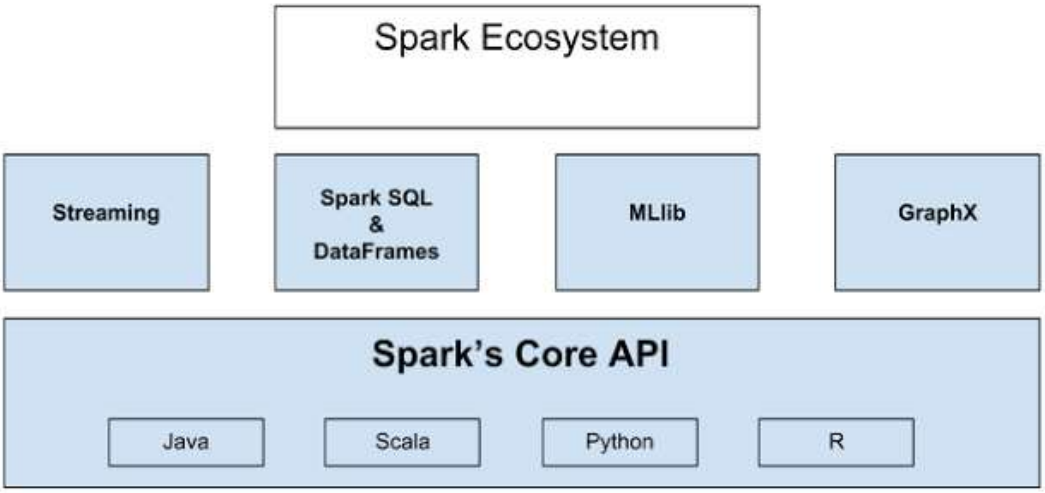
4.2 Apache Spark

Spark açık kaynak kodlu dağıtık büyük veri işleme sistemidir. Spark projesinin temelleri 2009 yılında Kaliforniya Berkeley üniversitesinde atılmıştır. 2010 yılında açık kaynak kodlu bir yazılım haline getirilmiştir. 2013 yılında ASF tarafından desteklemiştir. 2014 yılında kurumun zirve projesi haline gelmiştir. Projenin çok büyük bir bölümü Scala programlama diliyle yazılmıştır(Spark 2016).

Spark Hadoop'un kullanmış olduğu map-reduce modelinin eksiklerini iyileştirmeyi hedefler. Bu eksiklerden biri Hadoop üzerinde Hive ve Pig gibi SQL araçlarının sorgu cevaplarını geç getirmesinin önüne geçmektir. Bir diğeri ise makine öğrenmesi algoritmaların çalışma prensibi olarak bir veri kümesinin defalarca HDFS dosya sisteminden okunmasıdır(Zaharia ve diğ. 2010). Algoritma çalışırken defalarca aynı veriyi okumaya ihtiyaç duyabilir. Bu durumda sürekli olarak HDFS dosya sisteminden verini gelmesi beklenmektedir. Bu da çalışma süresinin uzamasına neden olmaktadır.

Spark kullanıcılarına Resilient Distributed Dataset (RDD) veri yapısını sunmaktadır. Bu veri yapısı küme üzerindeki node'lara dağıtılmış olarak bulunan ve objelerden oluşan sadece okuna bilen veri kümeleridir(Apache Spark 2016). Bu veri kümeleri hata tolerans sağlamaktadır. Bu veri kümelerinde bir kayıt olduğunda yeniden otomatikman üretilmektedir. RDD map-reduce modelindeki lineer veri akışına karşı geliştirilmiştir. RDD klasik map-reduce yönteminin aksine paylaşılabilir bellek üzerinde çalışmaktadır. Böylece verinin okunması için harcanan süre azaltılmıştır.

Şekil 4.6'da Apache Spark'ın ekosistemi gösterilmiştir. Spark'ın çekirdek yapısında birçok API mevcuttur ve bu API'ler birden fazla programlama diline destek vermektedir.



Şekil 4.6: Spark ekosistemi (Spark, 2016)

Ekosistemdeki streaming yapısı sürekli akan veriler üzerinde analiz yapmaya olanak sağlamaktadır(Zaharia ve diğ. 2010). Sürekli akan verilere örnek olarak log bilgileri veya sensörlerden gelen bilgiler verilebilir. Bu bilgileri dosya sistemine kaydedilmeden analiz edilmesi önemlidir. Uygulama olarak savunma sanayideki görüntü ve sensor bilgileri verilebilir. Spark streaming özelliği ile buna olanak sağlamaktadır.

Spark SQL yapısal verileri veya yarı yapısal verileri işlemeye olanak sağlar(Apache Spark 2016). Hadoop'ta kullanılan Hive teknolojisine benzer şekilde SQL sorgulama dilini destekler.

MLlib, Spark'ın çekirdek yapısı üzerinde çalışan dağıtık manine öğrenmesi çatısıdır. Sınıflandırma, regresyon analizi ve kümeleme algoritmalarının birçoğunu içermektedir. Bu algoritmalar dağıtık olarak çalışacak şekilde iyileştirilmiştir.

GraphX Spark üzerinde dağıtık graph işleme sağlayan modüldür. Graph işlemeyle ilgili birçok servis sağlamaktadır.

4.3 Apache Flink

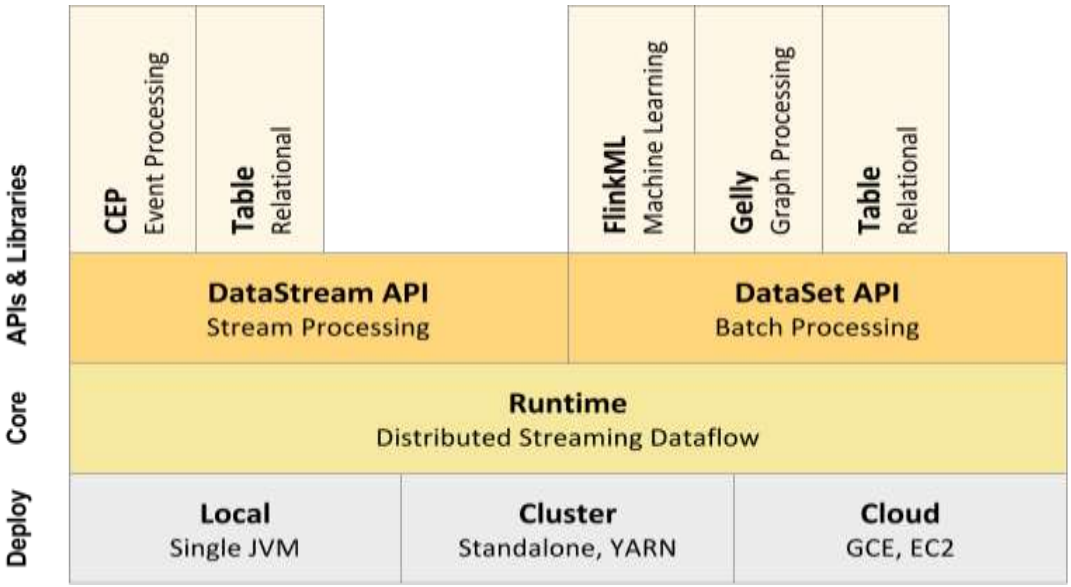
Hadoop ve spark'tan sonra büyük veriyi işlemek için geliştirilen bir diğer proje de Apache Flink olmuştur. Flink dağıtık açık kaynak kodlu büyük veri işleme aracıdır.

Apache Flink Java ve Scala programlama dilleriyle yazılmış sürekli akan büyük veriyi dağıtık işleme aracı olarak tasarlanmıştır(Flink 2016).

Flink projesinin temelleri 2010 yılında Almanya'da atılmıştır. 2014 yılı mart ayında ASF tarafından desteklenmiştir. 11 Mayıs 2016'da Apache Flink 1.0.3 sürümü kullanıma sunulmuştur(Flink-3 2016).

Flink'in Spark'a göre hafıza yönetimi, akan veri işleme hızı gibi üstünlükleri mevcuttur. Spark veri işlemede toplu işleme için geliştirilmiş ve akan veri işleme için de kullanılabilmekteyken Flink direk olarak akan veri işleme için geliştirilmiştir. Ayrıca Flink hafıza yönetimini kendisi yapmaktadır(Flink-2 2016).

Şekil 4.7'de Flink'in genel mimarisi gösterilmiştir. Flink en alt katman da birden fazla seçenek sunmaktadır. Local veya bir küme sisteminin üzerine kurulabilmektedir. Çekirdek katmanında veri işleme işlerinin yapıldığı yapı mevcuttur. Bir üst katmanında ise akan veri ve toplu veri işleme işlemlerini gerçekleştirmektedir. Spark veya Hadoop'a benzer şekilde en üst katmanında kullanıcılar birçok API sunmaktadır. Table API'siyle SQL gibi ifadelerinin yazılabilmesine olanak sağlamaktadır. FlinkML ile makine öğrenmesi algoritmalarının kullanılmasına olanak sağlamaktadır. Spark'a benzer şekilde Gelly kütüphanesiyle de graph işlemlerinin yapılmasına altyapı sağlamaktadır.



Şekil 4.7: Apache flink (Flink-3, 2016)

5. VERİ MADENCİLİĞİ

Bilginin insanın olduğu her yerde her an üretiliyor olması, bu bilginin saklanması yönetilmesi gibi zorunlulukların doğmasına neden olmuştur. İnsanlar bu zorunlulukları yerine getirmek için tarihsel süreçte yeni algoritmalar yeni yazılımlar geliştirmişlerdir. İlk olarak 1960 yıllarında veriler dijital olarak bilgisayar ortamında saklanmaya başlanmıştır. 1970 yıllarında bu veriler ilişkisel veri tabanı sistemlerinde depolanmaya başlanmıştır. 1980 yıllarında ise SQL dili yardımıyla bu veriler üzerinde sorgulamalar yapılmıştır. Bu sorgulamalar veriler içerisindeki bilgileri anlamlı dizilimler(cümleler) olarak kullanıcılara sunmaktaydı. Bu gelişme veri madenciliği disiplinin ortaya çıkmasındaki etkenlerden biri olarak düşünülebilir. Yazılan sorgular ile veri tabanlarında saklanan veriler arasında ilişkiler baz alınarak verilerin parçaları birleştiriliyordu. Bu sorgular basit soruların cevaplarından oluşmaktadır. Örneğin bir market veri tabanı için bu sorular;

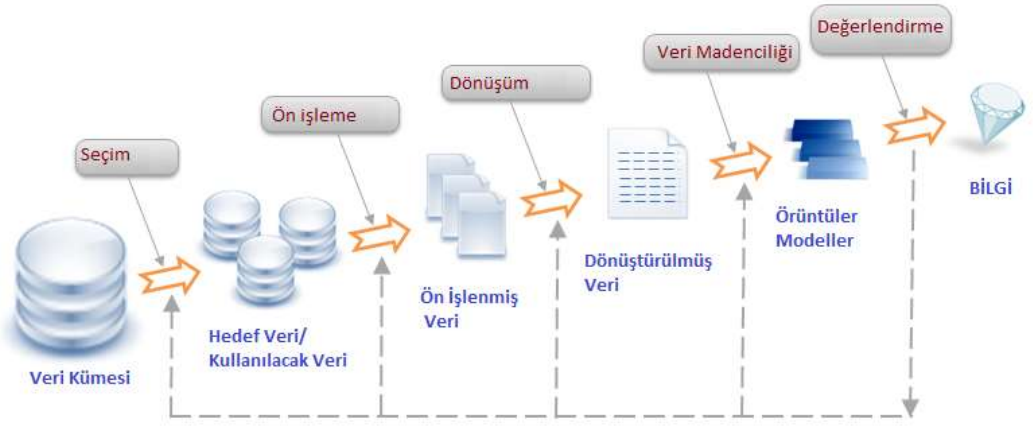
- Stoktaki A ürünün sayısı?
- Stoktaki B ürünün fiyatı?
- En çok satılan ürünün maliyeti?

Buna benzer sorguların kullanılması pazarlama, gelecek tahminleri ve raporlama gibi birçok avantaj sağlamıştır. Bu avantajların yanı sıra insanlar daha büyük beklenti içine girmişlerdir. Bu beklentileri sorguların ötesine geçmek, ilk bakışta sorgularla cevabı alınamayan sorulara cevap bulmak arayışı olmuştur. Bu arayışın sonucu veri madenciliği bilimi ile gerçekleştirilmiştir. Veri madenciliğinde market veri tabanı için aşağıdaki sorulara benzer soruların cevaplarının bulunması amaçlanmıştır.

- Benzer alım davranışı gösteren müşteriler?
- A ürünü ile birlikte en az satın alınan ürün hangisidir?
- B ürünün fiyatının artması A ürünün satışını nasıl etkiliyor?

1989 yılında ilk kez Gregory Piatetsky Knowledge Discovery in Databases(KDD)(Veritabanlarında Bilgi Keşfi) isimli bir çalışma düzenlemiştir. Bu çalışmanın isimi 1995 yılında Knowledge Discovery and Data Mining olarak değiştirilmiş ve yıllık olarak yapılmaya başlanmıştır(History Data Mining 2016). Bu

çalışma veri madenciliği biliminin temellerini oluşturmaktadır. "Veri tabanlarında Bilgi Keşfi" kavramı veri tabanlarında saklanan veri kümelerinden faydalı bilgilerin çıkarılması sürecidir(KDD 2016). Bu süreç 5 temel aşamadan oluşmaktadır ve bu aşamalar Şekil 5.1'de gösterilmiştir. Bu aşamaların bir adımı da veri madenciliğidir. 1995 yılından günümüze veri madenciliği disiplini çok çalışılmış ve kullanılmıştır. Şekil 5.1'de de görüldüğü gibi süreçteki birinci adımdan sonraki her adımdan önceki adımlara geçişler vardır. Bu geçişlerin nedenleri bir adımdaki işlemin istenilen kalitede/doğrulukta olmaması durumunda adımın veya daha önceki adımların tekrar gerçekleştirilebiliyor olmasıdır. Bu şekilde sürecin bir adımı birden fazla kez tekrar edilebilir.



Şekil 5.1: Veri tabanlarında bilgi keşfi süreci (KDD, 2016)

Veri tabanlarında bilgi keşfi sürecin seçim aşamasında bir veri tabanından veya veri ambarından üzerinde çalışılmak istenen veriler belirlenir. Veri belirleme işlemi ulaşılmak istenen amaç doğrultusunda gerçekleştirilir. Yani büyük bir hastane veri tabanını düşündüğünüzde bu veri tabanından hangi tabloların seçileceği bu aşamada belirlenir. Benzer bir şekilde veri ambarından hangi veri tabanlarının seçileceği bu aşamada belirlenir. Veri kümesinin belirlenmesinde çalışmanın hangi amaç doğrultusunda yapılacağı, hangi sorunun cevabının bulunmasıyla alakalıdır. Seçim aşamasından sonra hedef veri kümesi bir alt veri tabanı veya birkaç tablo olabilir. Bu aşamada amaç sorgulanabilir örneklem veri kümesi oluşturmaktır.

Sürecin önişleme aşamasında seçilmiş olan veri kümesi üzerinde hatalı kayıtların düzeltilmesi, silinmesi veya bütünleştirilmesi gibi işlemler gerçekleştirilir. Benzer şekilde eksik alanların(kayıtların) doldurulması söz konusudur(Silahtaroğlu

2013). Eksik veya hatalı kayıtlar sürecin ilerleyen aşamalarda uygulanacak olan veri madenciliği algoritmaların başarımını düşürmektedir. Bu nedenle önişlem aşaması önemli bir basamaktır. Bütün süreç adımları göz önüne alındığında önişlem aşamasının en çok çaba gerektiren adım olduğu söylenebilir.

Dönüşüm aşamasında önişlemden geçmiş olan veri kümesi üzerinde indirgeme işlemleri gerçekleştirilir. Kullanılacak olan algoritmaya göre bazı alanlar belli veri türüne dönüştürülmesi gerekebilir. Algoritmalardan bazıları metin alanlar üzerinde çalışır bazıları ise sayısal alanlar üzerinde çalışır(Döven 2013). Bu nedenle bazı alanlar için dönüşüm işlemleri gerçekleştirilir.

Veri madenciliği veri tabanlarında bilgi keşfi sürecinin en önemli aşamasıdır. Veri madenciliğinin literatür de birçok tanımı mevcuttur. Veri madenciliği varlığı daha önceden bilinmeyen anlamlı bilgilerin veri tabanlarından veya veri ambarlarından çıkarılması ve daha sonra karar vermek için kullanılmasıdır(Silahtaroğlu 2013). Veri madenciliği mevcut verilerden faydalı olabilecek verilere erişilmesidir(Döven 2013). Bir başka tanım ile veri madenciliği üretilen veri kümeleri arasından belli yöntemlerle faydalı bilgileri ortaya çıkarma sürecidir(Ergün 2012). Veri madenciliği büyük veri kümeleri içerisinde bulunan gizli olan örüntüleri, ilişkileri ortaya çıkarma işleminde istatistik veya yapay zekâ gibi bilim alanlarının kullanıldığı bir süreçtir(Dolgun ve diğ. 2009).

Yapılan tanımlardan da anlaşılacağı üzere büyük boyuttaki veri tabanları veya veri ambarlarından anlamlı, kullanılabilecek faydalı bilgilerin veya ilişkilerin ortaya çıkarılması ve bu anlamlı bilgilerin karar verme aşamalarında kullanılması sürecidir. Bu bilgilerin ortaya çıkarılması aşamasında makine öğrenmesi algoritmaları veya istatiksel yöntemler kullanılmaktadır. Bir bütün olarak veri madenciliği konsepti ve veri madenciliği alanında kullanılan ilişkili disiplinler Şekil 5.2'de gösterilmiştir.

Şekil 5.2: Veri madenciliği konsepti

Veri tabanında bilgi keşfi sürecinin son aşaması olan değerlendirme aşamasında, veri madenciliği aşamasından elde edilen çıktıların son kullanıcılar tarafından anlaşılabilir bir formatta sunulmasını hedefler. Bu çıktılar görsel olarak grafiksel, ağaç modelleriyle, tablolarla veya sayısal verilerle ifade edilir. Çıktıların iyi ve doğru sunulması bilgi çıkarım sürecinin başarımını doğrudan etkileyeceğinden, bu aşamanın dikkatli gerçekleştirilmesi gerekmektedir. Çıktıların doğru değerlendirilmesi veya yorumlanması da tüm sürecin başarımını etkileye bilmektedir.

Temelde veri madenciliği yöntemleri sınıflandırma(Classification), kümeleme(Clustering) ve birliktelik kuralları(Association Rules) şeklindedir(Coenen 2004). Sınıflandırma gelen yeni bir kaydın özellikleri değerlendirilerek mevcut olan sınıflardan hangisine dâhil edileceği sorusuna cevap arayan yöntemdir. Sınıfı belirlenecek kaydın özelliklerinin değerlendirilmesi için sınıflandırma algoritmaları için eğitim verisi hazırlanır. Eğitim verisi ile eğitilen algoritma sonucunda model elde edilir, bu model yardımıyla hangi sınıfa ait olduğu bilinmeyen kayıtlar özelliklerine göre sınıflara dağıtılır.

Kümeleme yönteminde bir veri kümesinin sınıfları belirlenir. Kümelemede sınıfları belirsiz olan veri kümeleri içerisindeki elemanların benzerlikleri veya bazı özellikleri referans alınarak bu elemanların gruplandırılması işlemidir. Verilerin kaç kümeye ayrılacağı veya verilerin hangi kümeye ait olacağını verilerdeki benzerlikler belirlemektedir(Silahtaroğlu 2013). Örneğin elimizde boy ölçüleri ve başka bilgileri bulunan 10 kişiyi bir basketbol ön elemesi için kümelemek istediğimizi varsayalım. Kişilerin boy ölçüleri Tablo 5.1'de gösterildiği gibidir.

Tablo 5.1: Kişi boy ölçüleri

Kişi Adı	**Boyu**
Ahmet	1.88
Ali	1.92
Kemal	1.70
Cemal	1.74
Selim	1.73
Yaşar	2.01
Ufuk	2.04
Hasan	1.73
Hüseyin	1.89
Gökhan	1.99

Bu kişileri boy ölçülerine göre kümelemek istediğimizde, boy ölçüleri incelendiğinde 1.72 civarında bir yığılma, 2.0 civarında bir yığılma ve 1.9 civarında bir yığılma olduğu söylenebilir. Bu durumda boy ölçüleri benzerlikleri referans alınarak bu kişiler üç kümeye ayrılabilir. Küme isimleri uzun, orta ve kısa olarak belirlenirse kümeleme sonucu Tablo 5.2'deki gibi olacaktır.

Tablo 5.2: Kümeleme sonucu

Küme Adı	**Kişiler**
Uzun	Gökhan, Yaşar, Ufuk
Orta	Ahmet, Ali, Hüseyin
Kısa	Hasan, Kemal, Cemal, Selim

Bir diğer veri madenciliği yöntemi olan birliktelik kuralları ise veri tabanında saklanan birçok kaydın diğer kayıtlarla olan bağlantılarını açıklayan işlemler topluluğudur(Silahtaroğlu 2013). Bağıntıları kurallar olarak ifade edildiğinden birliktelik kuralları yöntemi denilmektedir. Yöntemin amacı bir veri tabanı içerinde eş zamanlı olarak gerçekleşen olaylar arasındaki bağıntıları ortaya çıkarmaktır. Birçok kullanım alanı olmasına rağmen en çok pazarlama alanında kullanıldığından sepet analizi yöntemi olarak literatüre geçmiştir. Pazarlama alanında müşterilerin davranışlarının belirlenmesinde kullanılmaktadır(Coenen 2004). Pazarlama alanında uygulamalarından biri olarak, kola ve çekirdek alan bir müşterinin aynı zaman da %80 oranında cipste satın aldığı bilgisinin ortaya çıkarılmasında kullanılmaktadır.

Veri madenciliği teknikleri ve teknolojilerinin gelişmesinin sonucu olarak günümüzde veri madenciliği birçok alanda kullanılmaktadır. Sağlık alanından finans alanına, eğitimden eğlence alanına kadar geniş bir uygulama alanı bulmuştur. Bu alanlardan bazıları aşağıda sıralanmıştır(Döven 2013);

- Bankacılık
  - ✓ Kredi Hesaplama
  - ✓ Dolandırıcılık Tespiti
  - ✓ Kredi Taleplerinin Değerlendirilmesi
  - ✓ Risk Analizleri
- Genetik
  - ✓ Hastalık Belirleme
  - ✓ Gen Bozuklukları Tespiti
- Pazarlama
  - ✓ Pazar Sepet Analizi
  - ✓ Satış Tahmini
  - ✓ Müşteri İlişkileri Yönetimi
  - ✓ Müşteri Gruplandırılması
- Eğlence/Müzik
- Şans Oyunları
- Sigortacılık
- Eğitim
- Kamu Kurumları

- ✓ Yatırım
- ✓ Hisse Senetleri

- Spam e-posta Belirleme
- İmalat
- Tıp
  - ✓ Hastalık Teşhisi
  - ✓ Tedavi Yöntemleri Belirlenmesi
- Bilim

Günümüzde sınıflandırma, kümeleme ve birliktelik kuralları için kullanılan birçok algoritma mevcuttur. Bu algoritmalardan tez çalışmasında sınıflandırma için kullanılan Naive Bayes ilerleyen bölümlerde anlatılacaktır.

5.1 Sınıflandırma Algoritmaları

Sınıflandırma mevcut kayıtları referans alarak gelecekteki kayıtların sınıflarını belirleme yöntemidir. Mevcut kayıtların hangi sınıfa ait olduğu önceden belirlenmelidir. Bu durumda sınıflandırma algoritmaları için eğitim verisi hazırlamak gerekebilir. Eğitim verisi algoritmaya önceden hangi özellikteki kayıtların hangi sınıfa ait olduğunu gösteren alt veri kümesidir. Bu veri kümesinin doğru belirlenmesi ve hatalardan arındırılmış olması algoritmanın başarımını yükseltecektir. Eğitim verisinin yanında bir de test verisi hazırlanmalıdır. Test verisi eğitim verisine benzer şekilde kullanıcı denetimi ile hangi kayıtın hangi sınıfa ait olduğundan oluşan alt veri kümesidir. Eğitim verisi ile eğitilen algoritma çıktı olarak öğrenilmiş bir model üretmektedir. Bu model daha sonra sınıflandırılmak istenen veri kümesine uygulanmakta ve sınıfı belli olmayan veri kümesindeki kayıtlar sınıflandırılmış olur. Algoritmanın başarımını test etmek için oluşturulan test veri kümesinin algoritmanın oluşturduğu model tarafından sınıflara ayrılması istenir, bu şekilde sınıflandırma işleminin başarı kriterleri görülebilir.

Sınıflandırma yöntemi günümüzde birçok alanda kullanılmaktadır. Zararlı e-postaların tespit edilmesinde, web sunucularına yapılan saldırıların tespit edilmesinde, dolandırıcılık tespitinde, ürünlere ait müşteri yorumlarından müşteri memnuniyetlerinin belirlenmesinde sınıflandırma algoritmaları kullanılmaktadır. Bu

tez çalışmasında da gazetelere ait tweet'ler Naive Bayes sınıflandırıcı yardımıyla sınıflandırılacaktır.

5.1.1 Naive Bayes Sınıflandırıcı

Naive bayes algoritması olasılık hesaplamalarına dayanan bir istatiksel sınıflandırma yöntemidir(Erten, 2015). Bayes algoritması mevcutta var olan ve belli bir sınıfa ait olan verileri kullanarak yeni verinin hangi sınıfa ait olacağını belirler(Silahtaroğlu 2013). Bayes algoritması bir kayıt için bir dizi olasılık hesaplamaları gerçekleştirmektedir. Eğitim verisinin yani mevcut kayıtların sayısının fazla olması bayes algoritmasının yavaş çalışmasına neden olmaktadır.

Naive bayes teoremine göre sınıflandırıcı Denklem 5.1'deki formül yardımıyla sınıflandırma yapmaktadır.

$$P\,(\mathrm{A}\,|\mathrm{B}) = \frac{\mathrm{P}(B\ |A)*P(A)}{P(B)} \tag{5.1}$$

P(A |B); B olası durumunda A durumunun oluşma olasılığıdır.

P(B |A); A olası durumunda B durumunun oluşma olasılığıdır.

P(A) ve P(B); A ve B olaylarının ön olasılığıdır(Kuzucu 2015).

Bir örnek üzerinde Naive Bayes algoritmasının nasıl çalıştığını inceleyelim, örneğimizde elimizde bir eğitim verisi olacaktır. Bu eğitim verisi ile bir öğrenme modeli oluşturulacaktır. Daha sonra sınıfı belli olmayan bir kayıtın hangi sınıfa ait olacağı belirlemeye çalışılacaktır. Sınıf olarak olumlu ve olumsuz sınıfı mevcuttur. Eğitim verisi Tablo 5.3'te gösterildiği gibidir. Her bir cümle bu tez çalışmasında kullanılacak olan gazetelerin paylaşmış olduğu tweet'lerden oluşmaktadır. Test cümlesi ise "şiddetli fırtına sonrası öğrenci müjde tatil" şeklindedir. Cümlelerdeki ekler temizlenmiştir.

Tablo 5.3: Naive bayes eğitim verisi

Atılan Tweet	Boşluklara göre ayrılmış kelimeler				Sınıfı
1	personel	öğrenci	tatil	müjde	olumlu
2	personel	müjde	zam		olumlu
3	müjde	öğrenci	birinci	tatil	olumlu
4	çarpıştı	şiddetli	fırtına	ölü	olumsuz
5	ölü	öğrenci	şiddetli	çarpıştı	olumsuz

Öncelikle P(olumlu) ve P(olumsuz) değerleri hesaplanmak istendiğinde;

P(olumlu)=3/5=0,6 ve P(olumsuz)=2/5=0,4 şeklindedir.

Burada P(olumlu) eğitim verisindeki olumlu sınıfında bulunan kayıtların tüm kayıtlara oranı, P(olumsuz) ise olumsuz kayıtların sayılarının eğitim verisindeki tüm kayıtlara oranı şeklinde hesaplanmaktadır. Her kelimenin hangi sınıfa ait olduğuna dair koşullu olasılıkları aşağıdaki Denklem 5.2 ile hesaplanmaktadır.

$$P(A|B)=\frac{\left(B\ sınıfındaki\ kayıtlarda\ \text{“}A\text{”}\ kelimesinin\ tekrar\ sayısı\ +1\right)}{\left(B\ sınıfındaki\ kayıtlardaki\ tüm\ kelimelerin\ sayısı\ +\ Öğretilen\ veri\ sayısı\right)} \quad (5.2)$$

Olumlu sınıfı için kelimelerin koşullu olasılıkları hesaplandığında aşağıdaki gibi olacaktır.

P(öğrenci |olumlu)=(2+ 1)/(11+10)=3/21=0,1428

P(müjde |olumlu)=(3 + 1)/(11+10)=4/21=0,1904

P(personel |olumlu)=(2 + 1)/(11+10)=3/21=0,1428

P(zam |olumlu)=(1 + 1)/(11+10)= 2/21=0,0952

P(tatil |olumlu)=(2 + 1)/(11+10)=3/21=0,1428

P(birinci |olumlu)=(1 + 1)/(11+10)= 2/21=0,0952

Olumsuz sınıfı için kelimelerin koşullu olasılıkları hesaplandığında aşağıdaki gibi olacaktır.

P(çarpıştı | olumsuz)=(2 + 1)/(8+10)=3/18=0,1666

P(şiddetli |olumsuz)=(2 + 1)/(8+10)=3/18=0,1666

P(fırtına |olumsuz)=(1 + 1)/(8+10)=2/18=0,1111

P(ölü |olumsuz)=(2 + 1)/(8+10)=3/18=0,1666

P(öğrenci |olumsuz)=(1 + 1)/(8+10)=2/18=0,1111

Her iki sınıf için kelimelerin koşullu olasılıkları hesaplandıktan sonra test verisi için sınıfsal olasılıklar hesaplanabilir. Ancak test verisinde olup eğitim verisinde olmayan kelimeler mevcut, bu kelimeler eğitim verisinde bulunmadıklarından hesaplanmış bir sınıfsal olasılık değeri bulunmamaktadır. Bulunmadığı içinde kelimenin olasılık değeri 0 kabul edilemez, sonucu değiştirmemesi açısından çarpan olarak bu değer 1 kabul edilecektir(Kuzucu 2015). Test verisi için sınıfsal olasılıklar aşağıdaki gibi hesaplanmaktadır. Test verisi ”şiddetli fırtına sonrası öğrenci müjde tatil” şeklindeydi.

Olumlu sınıfı için;

P(olumlu |Test) = P(olumlu) * P(şiddetli | olumlu) * P(fırtına | olumlu) * P(sonrası | olumlu) * P(öğrenci | olumlu) * P (müjde | olumlu) * P (tatil| olumlu)

P(olumlu |Test)= 0,6*1*1*1* 0,1428 * 0,1904 * 0,1428 =0,0023 şeklindedir.

Olumsuz sınıfı için;

P(olumsuz |Test) = P(olumsuz) * P(şiddetli | olumsuz) * P(fırtına | olumsuz) * P(sonrası | olumsuz) * P(öğrenci | olumsuz)* P(müjde | olumsuz)* P(tatil| olumsuz)

P(olumsuz |Test) =0,4* 0,1666* 0,1111*1 *0,1111*1*1=0,0008 şeklindedir.

P(olumlu |Test) ve P(olumsuz |Test) olasılık değerleri incelendiğinde, sınıflandırmak istediğimiz bu yeni cümleyi algoritma olumlu sınıfındaki olasılık değeri büyük olduğundan olumlu olarak işaretleyecektir. Örnekte de görüldüğü üzere bir veri sınıflandırılacağı zaman her sınıf için olasılık değeri hesaplanır, veri en büyük olasılık değerine sahip sınıfa atanır.

Yapılan hesaplamadan da görüldüğü üzere eğitim verisindeki bir kelimenin değiştirilmesi(eklenmesi veya silinmesi) algoritmanın her kelime için olasılık

değerlerinin yeniden hesaplanmasını gerektirecektir. Bir dizi matematiksel hesaplamalar yaptığından algoritmaya terabaytlar düzeyinde eğitim verisinin giriş olarak verilmesi, algoritmanın hesaplamam süresini çok uzatacaktır. Büyük veri araçları bu hesaplama işlemi için dağıtık sistemler üzerinde paralel hesaplama yöntemleri geliştirmişlerdir. Bu çözümlerden biri de hadoop ve mahout kütüphaneleridir.

Veri madenciliğinin çok geniş bir kullanım alanı vardır. Kullanım alanın genişlemesi yapılan madenciliğin alanlara göre özelleşmesine neden olmuştur. Bu alanlar metin madenciliği, ses madenciliği, çizge(graph) madenciliği ve web madenciliği olarak sıralanabilir(Coenen 2004). Bu tez çalışmasında metin şeklinde saklanan dosyalar üzerinde metin madenciliği gerçekleştirilecektir.

5.2 Metin Madenciliği

Bilgisayar ortamında saklanan veriler çeşitli formatlarda olabilmektedir. Veri madenciliğinde kullanılan veriler, genellikle veri tabanı yönetim sistemlerinde saklanan yapısal verilerdir. Veri madenciliği yapısal veriler üzerinde gerçekleştirilmektedir. Yapısal veri, tablolarda satır ve sütun düzeninde saklanan verilerdir(Dolgun ve diğ. 2009). Metin madenciliği ise yapısal olmayan metin formatındaki veriler üzerinde anlamlı ve faydalı olabilecek bilgilerin ortaya çıkarılmasıdır(Oğuzlar 2011). Yapısal olmayan veriler e-postalar, kitaplar, dergiler, web sayfaları, XML dosyaları, resimler ve benzeri kaynaklardaki verilerdir. Metin madenciliği bu kaynaklardan metin formatında olanlardan önceden bilinmeyen ve yeni olan bilgilerin ortaya çıkarılması sürecidir. Bir diğer söylem ile metin madenciliği veri madenciliği sürecinin metin formatındaki veriler üzerindeki gerçekleştirilmesidir(Beyhan 2014). Metin madenciliğindeki amaç bir metin içerisindeki anlamın ortaya çıkarılması ve kullanılmasıdır.

Metin madenciliğinin veri madenciliğinin özelleşmiş bir alt disiplini olarak kabul edilebilir. Veri madenciliği sadece yapılandırılmış verilerle işlem yapmaktadır(Dolgun ve diğ. 2009). Metin madenciliğinde yapılandırılmamış olan veriler çeşitli işlemlere tabi tutularak yapılandırılmış formata dönüştürülür.

Dönüştürme işleminden sonra bu verilere veri madenciliği algoritmaları veya yöntemleri uygulanır.

Metin madenciliğinin günümüzde aktif olarak birçok alanda kullanılmaktadır. Bu alanlardan bazıları aşağıdaki gibidir(Karaca 2012).

- Müşteri ilişkileri yönetimi
- Sahtekârlık tespiti
- Sağlık
- Pazarlama
- Doküman özetleme
- Yazar tanıma sistemleri
- Web içerikleri sınıflandırma

Bunların haricinde aşağıdaki gibi kullanım alanları da mevcuttur.

- Doküman sınıflandırma(Aşlıyan ve Günel 2010)
- Benzer içerikleri belirleme(Karadağ ve Takçı 2010)
- Soru-cevap sistemleri(İlhan ve diğ. 2008)

Metin madenciliğinde veri madenciliğinde benzer bir süreç vardır, bu süreç metinlerin toplanması, metinlerin önişlemden geçirilmesi, metinlerin özellik vektörleri olarak ifade edilmesi, makine öğrenmesi veya istatistik tabanlı bir algoritma ile işlenmesi ve sonuçlarının değerlendirilmesinden oluşmaktadır.

Metinlerin önişlemlerden geçirilmesi aşamasında yapısal olmayan metin dokümanlar yapısal formata getirilmeyi amaçlar. İyi bir metin madenciliği için iyi bir önişleme süreci gerekmektedir. Metin önişleme de metinin türüne ve içeriğine bağlı olarak birçok teknik kullanılmaktadır, bu tekniklerin bazıları Tablo 5.4 ‘teki gibidir (Oğuzlar 2011). Bu tekniklerden gövdeleme işlemi bu tez çalışması kapsamında metin halinde saklanan tweet’ler üzerinde uygulanmıştır. Gövdeleme işleminde kelimenin köküne kadar inilmekte, çekim ekleri, iyelik ekleri vb. ekler çıkarılmaktadır.

Tablo 5.4: Metin madenciliği önişleme teknikleri

Teknik	Kullanım Amacı
İşaretleme	Dokümanların kısım, paragraf, cümle, kelime veya hecelere bölünmesidir.
Gövdeleme	Aynı isim veya fiilden üretilen tüm kelimeleri tek bir kelimeymiş gibi işleme alınmasını sağlamaktadır. Genellikle kelimenin köküne kadar gövdeleme yapılmaktadır.
Çok Kelimeli Özellikler	Bir kelime yerine birden fazla kelimeden oluşan öbeklerin özellik olarak seçilmesini amaçlayan yöntemdir.
Kelime Anlam Belirsizliğini Giderilmesi	Cümle içindeki bir kelimenin hangi anlamı verdiğini belirlemeye çalışır. Türkçe için bu işlemi yapan bir araç bulunmamaktadır.
Sözlük Oluşturma	Dile özgü anlamsal sözlük oluşturmayı amaçlar.
Sözcük Türü Etiketleme	Kelimelerin bulunduğu bağlama göre uygun sözcük türü ile açıklanmasıdır. Kelimeleri kategoriye ayırır.
Öbek Tanıma	İsim, fiil ve zarf öbeklerin tespit edilmesini amaçlar.
Sözdizimsel Analiz	Cümledeki her kelimenin diğer kelimelerle olan bağlantılarını ortaya çıkarmayı amaçlar.

Metin madenciliği için toplanan dokümanlara yukarıda özetlenmiş metin önişleme teknikleri uygulandıktan sonra her kelimenin doküman üzerindeki etkisinin belirlenmesi için bir sonraki aşamada sözcük ağırlıklandırma yapılmalıdır. Ağırlıklandırma işlemi ile bir metine ait kelimelerin ne kadar sıklıkla geçtiği, metinde kaç farklı kelime geçtiği belirlenmektedir. Ağırlıklandırma işlemi kelimelerin doküman üzerinde yaratmış olduğu anlam veya duygunun etkisinin belirlenmesi için kullanılır. Literatür de birden fazla ağırlıklandırma tekniği vardır. Bu tez çalışmasında Sözcük Frekansı-Ters Doküman Ağırlıklandırması(tf-idf) yöntemi kullanılmıştır. Bu yöntem Mahout aracı tarafından parametre olarak sunulmaktadır.

5.2.1 Tf-Idf Ağırlıklandırma Yöntemi

Bu yöntem sözcük frekansı(tf) ve ters doküman frekansı(idf) yöntemlerinin birlikte kullanılmasıdır. Sözcük frekansı bir sözcüğün doküman içerisinde geçme sayısıdır(İlhan ve diğ. 2008). Bir sözcük bir dokümanda çok sayıda geçiyorsa o sözcük doküman üzerinde belirleyicidir. Burada tf belirlenirken her dokümandaki her sözcük için ayrı bir değer taşımaktadır. Tf yönteminde bir kelimenin frekansı sadece o dokümanda geçme sayısıyla belirlenir(Karaca 2012). Literatürde birçok tf hesaplama çeşidi vardır, bunlar arasından en çok kullanılan aşağıdaki gibidir.

$$tf\left(t,d\right)=1+\log\left(f_{t,d}\right) \tag{5.3}$$

d: d dokümanı

t: d dokümanında geçen t sözcüğü sayısı

Ters doküman frekansı(idf) ise bir terimin tüm dokümanlar arasında çok mu az mı geçtiğini belirlemeye yaramaktadır. Sözcüğün doküman içerindeki ağırlığının diğer sözcüklerle ve diğer dokümanlarla birlikte değerlendirilmesidir(Beyhan 2014). Bir sözcüğün bütün dokümanlarda geçmesini ağırlık olarak belirler. Bir sözcük sadece birkaç dokümanda geçiyorsa ilgili sözcüğün değeri yüksek, çok fazla dokümanda geçiyorsa değerini düşük olarak belirler(Karaca 2012). Tf yöntemine benzer şekilde birçok kullanımı mevcuttur bu kullanımlardan en çok uygulananı Denklem 5.4'teki gibidir.

$$idf\left(t,D\right)=\log\left(\frac{N}{\left|\left\{d\in D:t\in d\right\}\right|}\right) \tag{5.4}$$

d: d dokümanı

t: d dokümanında geçen t sözcüğü sayısı

D: bütün dokümanlar

N: bütün dokümanların sayısı

Tf-idf ağırlıklandırma yöntemi ise her iki yöntemin çarpımı olarak hesaplanmaktadır.

$$tfidf\left(t,d,D\right)=tf\left(t,d\right)xidf\left(t,D\right) \tag{5.5}$$

$$tfidf\left(t,d,D\right)=1+\log\left(f_{t,d}\right)x\log\left(\frac{N}{\left|\left\{d\in D:t\in d\right\}\right|}\right) \tag{5.6}$$

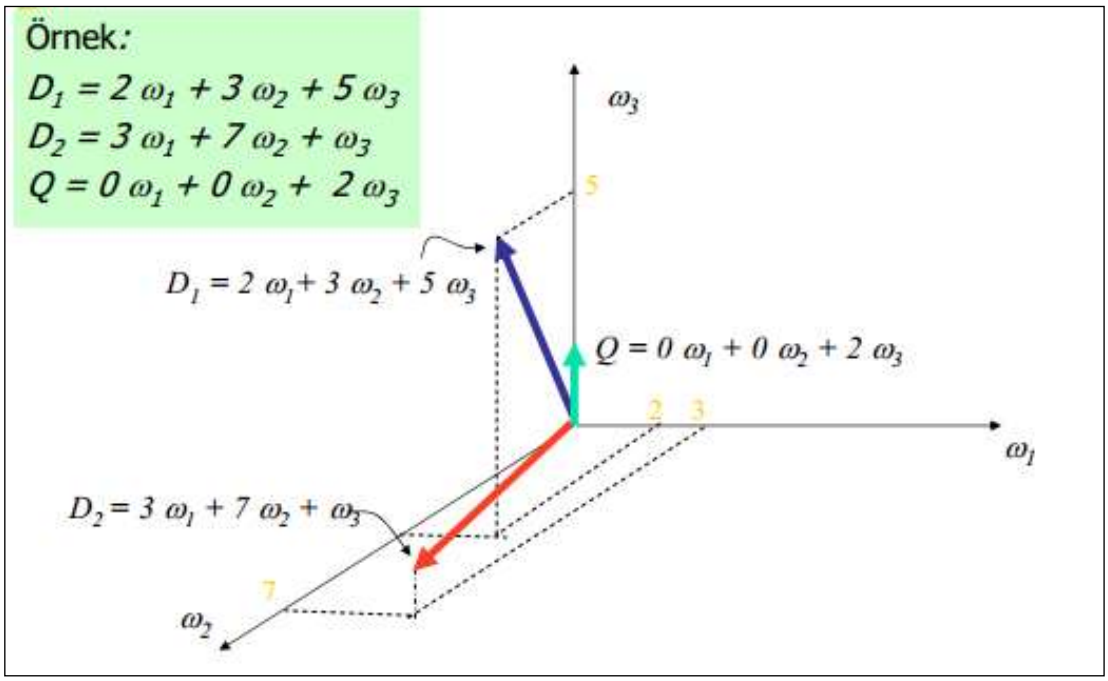
tfidf yöntemini aşağıdaki gibi değerlendirebiliriz(Karaca 2012);

- Bir sözcük farklı dokümanlarda çok geçmiyorsa sık geçen sözcüklere göre daha belirleyicidir.
- Her bir doküman için bir sözcüğün sık geçmesi onu daha belirleyici kılmaktadır.
- *tf* ile *idf* arasında ters orantı vardır.
- Bir dokümanda çok görülen yalnız diğer dokümanlarda az geçen bir kelimenin ağırlığı daha fazladır.

Bu tez çalışmasında sınıflandırma için kullanılan Apache Mahout kütüphanesi, metindeki kelimeleri ağırlıklandırma için parametre olarak verilebilen tf-idf yöntemini kullanmaya olanak sağlamaktadır.

5.2.2 Vektör Uzay Modeli

Metin ağırlıklandırma yöntemiyle metindeki kelimelerin sınıfsal dağılıma belirleyici etkisi belirlendikten sonra metinler sayısal formata getirilmelidir. Metin madenciliğinde veriler algoritmalar yardımıyla analiz edilme aşamasından önce, metinlerin sayısal formata dönüştürülmesi gerekir. Vektör uzay modeli dokümanı sayısal olarak ifade eder. Dokümanların gösterimi için en çok tercih edilen yöntem vektör uzay modelidir(Oğuzlar 2011). Bu yöntemde her bir metin belgesini sayısal bir model olan vektörlerle gösterimi sağlanır. Vektör uzay modeli ile yapısal olmayan metinler yapısal bir forma getirilmiş olacaktır.



Şekil 5.3: Vektör uzay modeli (Vektör uzay modeli, 2016)

i : 1,2,3,4 ….n, tam sayıyı ifade eder.

D_i : i'ninci dokümanı ifade eder.

Q : sınıflandırılmak istenen dokümanın vektörüdür.

ω_i : i'ninci terimi(sözcüğü) ifade eder.

Vektör uzay modelinin grafiksel gösterimi Şekil 5.3'teki gibidir. Sözcüklerin fazla olması modelin uzayda fazla eksene sahip olacağı anlamına gelmektedir. Vektör uzay modeli yardımıyla terim doküman matrisi oluşturulabilir. Terim doküman matrisi her bir vektörün bir satır olarak ifadesidir. Sütunlar kelimeleri ifade etmektedir. Örneğin aşağıdaki 5 cümleye ait terim doküman matrisi Tablo 5.5'teki gibidir;

D_1 : personel öğrenci tatil müjde

D_2 : personel müjde zam

D_3 : müjde öğrenci birinci tatil

D_4 : çarpıştı şiddetli fırtına ölü

D_5 : ölü öğrenci şiddetli çarpıştı

Tablo 5.5: Terim doküman matrisi örneği

	personel	**öğrenci**	**tatil**	**müjde**	**zam**	**birinci**	**çarpıştı**	**şiddetli**	**fırtına**	**ölü**
Doküman-1	1	1	1	1	0	0	0	0	0	0
Doküman-2	1	0	0	0	1	0	0	0	0	0
Doküman-3	0	1	1	0	0	1	0	0	0	0
Doküman-4	0	0	0	0	0	0	1	1	1	1
Doküman-5	0	1	0	0	0	0	1	1	0	1

Bir sonraki aşamada sayısal olarak vektörler ile ifade edilen dokümanların bir birine olan benzerlikleri hesaplanmalıdır. Vektörlerin benzerliklerinin hesaplanması için literatürde birçok yöntem bulunmaktadır ama en çok kullanılan yöntemler Jaccard veya Cosinüs benzerliği yöntemleridir.

Bu işlemlerden sonra metin madenciliğindeki metin formatındaki dosyalar veri madenciliği için kullanılabilecek formata getirilmiş olunacaktır. Birçok veri madenciliği aracı bulunmaktadır. Bu araçlardan bazıları metin dosyaları üzerinde işlem yapma özelliklerine sahiplerdir. Mahout bu özelliklere sahip olan araçlardan biridir. Kullanılacak yazılım aracı belirlendikten sonra veri kümesine normal veri madenciliği süreci uygulanabilir.

Metin madenciliği için önişleme aşamasında dokümanın içeriğinin ifade edildiği dilin yapısıyla ilgili işlemler gerçekleştirilmektedir. Bir dokümandaki kelimelerin özelliklerinin belirlenmesi, sonlandırıcı kelimelerin tespit edilmesi, aynı anlama gelen kelimelerin belirlenmesi gibi işlemler doküman dilinin yapısıyla ilgilidir. Metin madenciliği için metin dilinin özelliklerinin iyi bilinmesi, dildeki deyimlerin tespit edilmesi önemlidir. Bir metindeki ifadelerin, duyguların ortaya çıkarılıp sayısallaştırılması için günümüzde doğal dil işleme yöntemleri kullanılmaktadır. Bu çalışmada metin formatında saklanan tweet'ler üzerinde önce metin madenciliği süreçleri daha sonra doğal dil işleme yöntemleri uygulanacaktır.

5.2.3 Doğal Dil İşleme ve Duygu Analizi

Bilgisayar tarafından bir metnin insanların anladığı şekilde anlaşılmasının sağlanması konusu doğal dil işleme bilimi çalışma kapsamına girmektedir. Doğal dil işleme ile bir dilin morfolojik, yapısal ve sözdizimsel analizleri yapılmaktadır(Aktaş 2006). Bir cümledeki içeriğin insan beyninde yarattığı anlam ve algının bilgisayarlar tarafından anlaşılması, bilgisayar insan etkileşimi için çok önemlidir. Doğal dil işleme bu doğrultuda dili metin tabanlı veya ses tabanlı olarak ele almaktadır. Metin tabanlı olarak doğal dil işleme bir dilin sistematiğini formülize etmeyi amaçlamaktadır.

Günlük hayatta birçok doğal dil işleme gerçekleştirimi ile karşılaşmaktayız. Doğal dil işleme ile uzman sistemlerdeki soru cevap sistemleri bu alandaki çalışmalardan bir örnektir. Diller arası çeviri sistemleri benzer şekilde doğal dil işleme uygulama alanıdır. Sesli komutlarla yönetilen robotlar da aynı şekilde doğal dil işleme gerçekleştirimleridir. Bunların haricinde bir dille ilgili olarak dildeki cümlenin öğelerinin bulunması, kök ve eklerin belirlenmesi ve kelime tiplerinin belirlenmesi doğal dil işleme uygulamalarıdır(Altan ve Orhan 2005). Bir metinin kime ait olduğunu belirlemek doğal dil işleme uygulamalarından bir diğeridir.

Bir metindeki duygunun veya ifadenin bilgisayar algoritmaları yardımıyla ortaya çıkarılması bilgisayarların insanları anlamaya başladıkları anlamına gelmektedir. Bu beceri birçok alanda ufuk açan gelişmelere neden olabilir. Özellikle ticari olarak sosyal medyada paylaşılan ifadelerden içinde bulunulan duyguların, ruh halini tespiti önem arz etmektedir. Bu durum birçok alan için dinamik pazar stratejisi oluşturmaya olanak sağlayacaktır. Örneğin, sosyal medya paylaşımlardan bir kişinin ruh hali tespit edildiğinde, kişiye özel promosyonlarla harcama alışkanlıklarının arttırılması bu alanda yapılan çalışmalardan biridir. Benzer şekilde birçok kar amacı güden firma bu yöntemlerle insanlara daha hızlı erişmekte, insanların durumlarına göre hızlı bir şekilde ürün veya hizmet sunmaktadır.

Doğal dil işleme bir dil üzerindeki işlemler için makine öğrenmesi algoritmalarını kullanmaktadır. Özellikle sözcük anlamının tespit edilmesi yöntemlerinde makine öğrenmesi yöntemleri kullanılmaktadır. Doğal dil işleme için en çok kullanılan makine öğrenmesi algoritmalarına Naive Bayes, Karar Ağaçları, Random Forest örnek olarak verilebilir. Bu algoritmalar yardımıyla bir metindeki

ifadelerin, duyguların belirlenmesi çalışılabilmektedir. Doğal dil işleme ile analizi yapılan metinler içerisinde duyguların keşfedilmesi süreci duygu analizi olarak isimlendirilebilir. Duygu analizi ve doğal dil işleme birbirini tamamlayan disiplinlerdir. Bir metin içerisindeki cümlelerden kişinin özelliklerinin belirlenmesi bir duygu analizi çalışmasıdır. Duygu analizi yöntemiyle insanların bir konuda karar vermelerine yardımcı olunabilir. Bunun en belirgin örneklerinden biri, bir film hakkında yapılan izleyici yorumlarıdır. Bu sistem yardımıyla yapılan yorumlar değerlendirilip izleyiciye rapor olarak sunulursa, izleyici filmi izlemeye gitmeden önce filmin kalitesi hakkında fikir sahibi olabilir. Günlük hayatta genellikle bu işlemi manuel olarak film sitelerindeki yorumlar okunarak film hakkında karar vermeye çalışılır.

Bu tez çalışmasında da hem doğal dil işleme yöntemleri hem de makine öğrenmesi algoritmaları kullanılarak gazetelere ait tweet'lerdeki duygular tespit edilmeye çalışılacaktır. Doğal dil işleme yöntemleriyle metin formatındaki tweet'ler işlenecek, makine öğrenmesi algoritmalarından olan Naive Bayes ile metinler içerdikleri ifadelere göre olumlu veya olumsuz olarak sınıflandırılacaktır.

6. YAPILAN ÇALIŞMA

Büyük verinin işlenmesi günümüzün en popüler çalışma alanlarından biri haline gelmiştir. Büyük verinin işlenmesi yüksek kapasiteli sunucularda büyük hesaplamalar yapılmasından ibaret değildir. Bu verilerin içeriğinin değerlendirilmesi, faydalı sonuçlar elde edilmesi, büyük verinin işlenmesinin bir diğer önemli hususlarındandır. Bir veri kümesinden önceden sezinlenemeyen, varlığından habersiz olduğumuz anlamlı bilgilerin veya örüntülerin çıkarılması için veri madenciliği yöntemleri ve makine öğrenmesi algoritmaları kullanılmaktadır. Bu yöntemler yakın tarihe kadar yapısal veriler üzerinde kullanılmaktaydı. Büyük verinin doğası gereği veriler yapısal, yarı yapısal veya yapısal olmayan formatlarda bulunabilmektedir. Bu yapısal olmayan veriler üzerinde anlamlı ve faydalı örüntüler çıkarmak için büyük veri araçlarının yanında bu madenciliği yapacak yeni araçlar geliştirilmiştir. Apache Mahout bu araçlardan biridir.

Yapılan çalışma kapsamında büyük veri işleme araçlarından Hadoop tercih edilmiştir. Güçlü yapısı, açık kaynak kodlu olması, dosya sisteminin güçlü olması ve map-reduce modeliyle iyi performans sağlaması Hadoop'un tercih edilmesindeki bazı faktörlerdir. Hadoop'un kullanıcılarına sunmuş olduğu Mahout kütüphanesi, bazı makine öğrenmesi algoritmalarının map-reduce formatında yazılmış halini barındırmaktadır. Bu makine öğrenmesi algoritmaları veriler üzerinde sınıflandırma, kümeleme veya öneri gibi temel veri madenciliği yöntemlerini içermektedir. Bu tez çalışması kapsamında Mahout yardımıyla metin formatındaki veriler üzerinde sınıflandırma işlemi gerçekleştirilecektir.

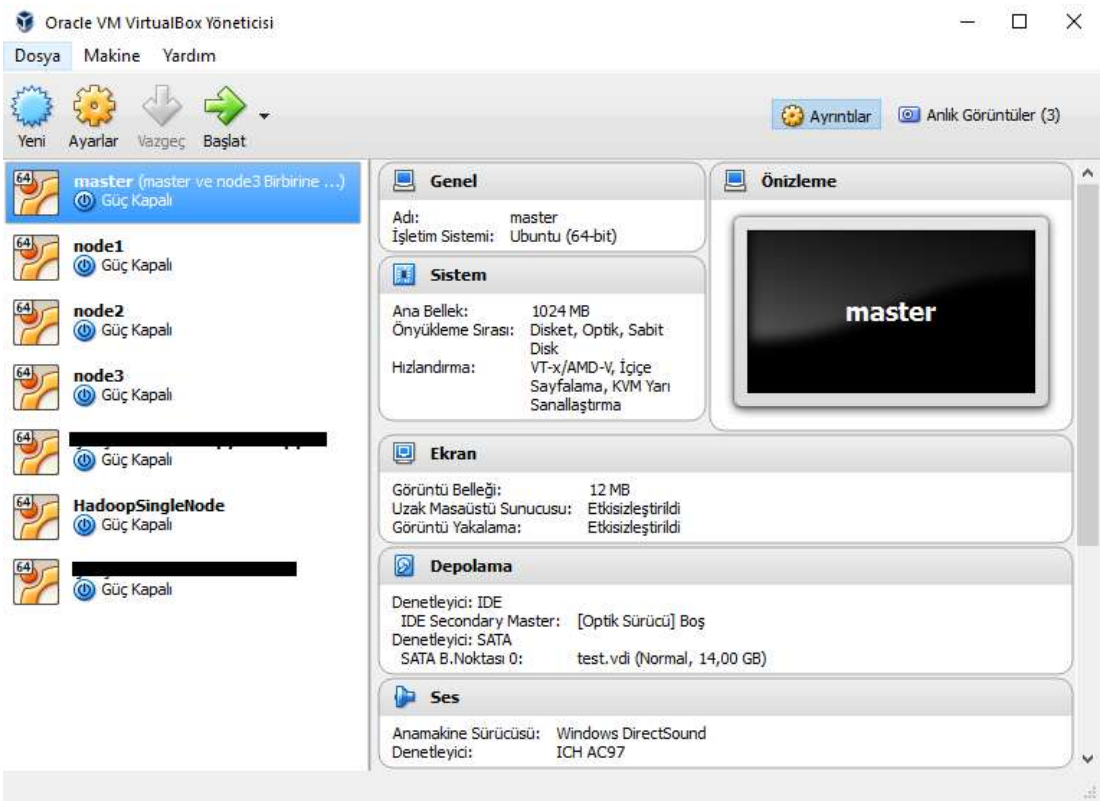
Sınıflandırma işlemi için Naive Bayes sınıflandırma algoritması kullanılmıştır. Sınıflandırma için 15 günlük gazetenin Twitter hesaplarında paylaşmış oldukları haberlerin başlıkları toplanmıştır. Yaklaşık 105.000 adet haber başlığı toplanmıştır. Bu haber başlıkları için bazı metin madenciliği ön işlemleri gerçekleştirilmiş ve veri kümesi oluşturulmuştur. Bu işlemler sonraki bölümlerde detaylıca anlatılacaktır.

Oluşturulan bu veri kümesi iki farklı Hadoop sistemi üzerinde çalıştırılmıştır. Hadoop'un iki farklı kurulumu(tek node'luk ve 4 node'luk küme) gerçekleştirilmiştir. Naive Bayes algoritması ve veri kümesi her iki Hadoop ortamı üzerinde çalıştırılmıştır. İki farklı ortam için kullanılan işletim sistemleri, java versiyonu, Hadoop versiyonu ve

Mahout versiyonu aynıdır. Farklı iki ortamda çalıştırılmasındaki amaçlardan bazıları, algoritmanın çalışma süresi, sınıflandırma başarım ölçütleri gibi değerlerin nasıl değiştiğini gözlemlemektir.

6.1 Alt Yapının Hazırlanması

Veri madenciliği için büyük veri araçlarından hadoop kullanılacaktır. Hadoop büyük veriyi dağıtık olarak işleyebilen, açık kaynak kodlu bir yazılım çatısıdır(Wikipedia Hadoop 2016). Hadoop'un kurulumu sanal bilgisayarlar üzerinde gerçekleştirilmiştir. Sanal bilgisayarların oluşturulması için açık kaynak kodlu Oracle VM VirtualBox(sürüm 5.0.10 r104061) programı kullanılmıştır(VirtualBox 2016). Şekil 6.1'de sanal bilgisayarların ekran görüntüsü mevcuttur.



Şekil 6.1: Sanal bilgisayarlar

Sanal bilgisayarlardan master, node1, node2 ve node3 isimli bilgisayarlar bir küme olarak çalışmaktadırlar. Her sanal bilgisayar için bir gigabayt ram bellek, bir işlemci ve 10 gigabayt depolama alanı tahsis edilmiştir.

Sanal bilgisayarların kurulumu için kullanılan bilgisayar; hp marka, windows 64-bit işletim sistemine sahip, 8 gigabayt ram bellek, 128 gigabayt SSD(Solid-State Disk) hard disk ve intel core i-7 @ 2.0 GHz işlemciye sahiptir.

Hadoop kurulumu için oluşturulan sanal bilgisayarlara üzerinde Ubuntu (versiyon14.04.3) işletim sistemi kurulumu gerçekleştirilmiştir. İşletim sisteminin kurulumundan sonra Hadoop kütüphanesinin çalışması için java'nın kurulması gerekmektedir. Her bilgisayara java kurulumu gerçekleştirilmiştir. Kurulan java versiyonu Java 1.7.0.85 şeklindedir. Bu tez çalışmasında sayfa sayısının artmaması ve yeni Hadoop kurulum araçlarının geliştirilmesi nedeniyle Hadoop kütüphanesinin adım adım kurulumu anlatılmayacaktır. Hadoop kütüphanesinin kullanılması için yapılması gereken temel işlemler aşağıdaki gibidir (Hadoop Kurulum 2016);

- Hadoop kütüphanesinin çalışacağı bilgisayar ve işletim sistemi belirlenmeli(Mac, Ubuntu, Fedora).
- İşletim sistemi güncellenmeli.
- Java kurulumu yapılmalı.

Bu aşamadan sonraki adımlar ubuntu işletimi üzerinde tek node'luk Hadoop kurulumu için anlatılmıştır.

- Hadoop'un çalıştırılması için bir Hadoop kullanıcısı oluşturulmalı.
- Hadoop kümedeki node'ları yönetmek için SSH(Secure Shell) protokolüne ihtiyaç duyar, bunun için SSH anahtarı oluşturulmalı, SSH makinelerin erişim yetkisine sahip bir şekilde kendi aralarında haberleşmesini sağlar.
- Kurulması istenen Hadoop versiyonu indirilir ve sıkıştırılmış dosyadan çıkartılır. Çıkarılan Hadoop klasörüne Hadoop kullanıcısı yetkisi verilir.
- Hadoop için yapılandırma dosyalarının dosya yolu tanımlamaları $HOME/.bashrc dosyasında saklanır. Bu dosyanın içine Hadoop klasörünün yolu ve kurulmuş olan java'nın dosya yolu yazılmalıdır.
- Hadoop ana klasörünün içerisindeki conf dizinin altındaki hadoop-env.sh dosyasının içerisine kurulmuş olan java'nın dosya yolu(/usr/lib/jvm/java-1.7.0.85 gibi) yazılmalıdır.

- Bu işlemlerden sonra Hadoop'un çalışırken kullanacağı klasör(tmp) oluşturulmalı ve klasöre Hadoop kullanıcının yetkisi verilmelidir.

Bu aşamadan sonra Hadoop yapılandırma ayarları yapılmalıdır. Yapılandırma ayarları için Hadoop klasörünün altındaki conf dizinin içerisindeki üç temel yapılandırma dosyası üzerinde ayarlamalar yapılmalıdır. Bu dosyalar;

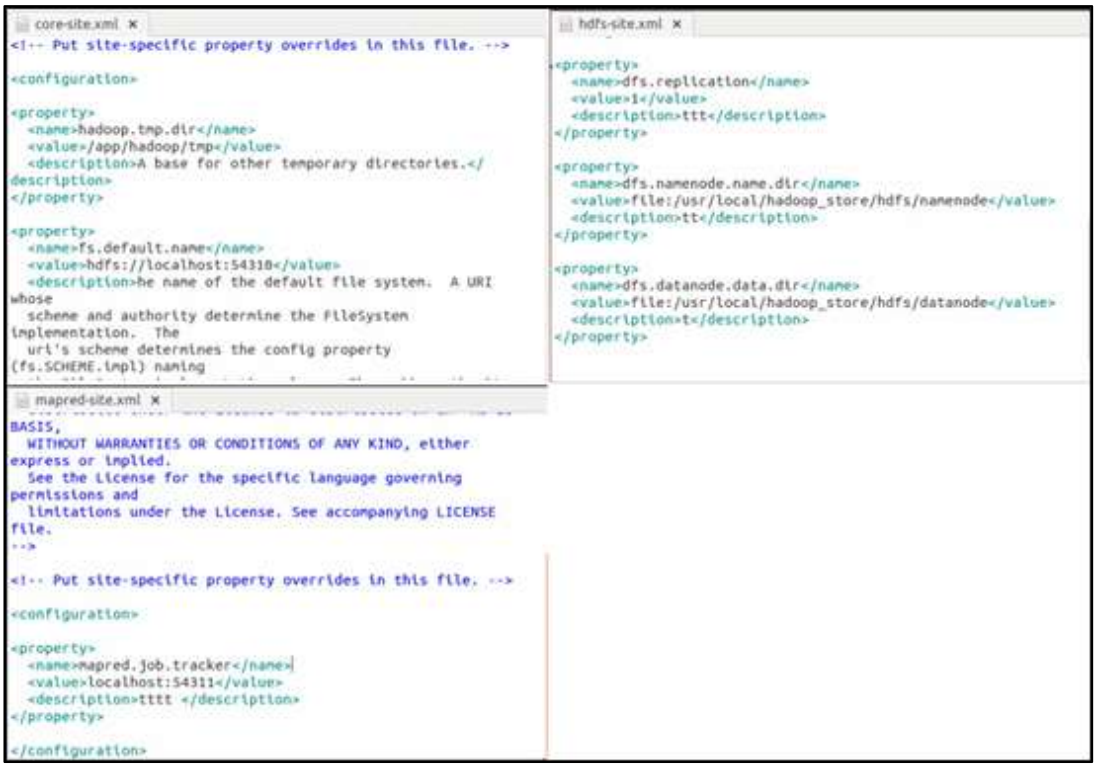
- core-site.xml
- mapred-site.xml
- hdfs-site.xml

şeklindedir. Bu dosyaların görevleri Tablo 6.1'de yazıldığı gibidir.

Tablo 6.1: Yapılandırma dosyaları ve görevleri

Dosyanın Adı	Görevi
core-site.xml	Hadoop'un dosyaları saklamak için kullanacağı dizinlerin yollarını ve dosya sisteminin kullanacağı host, port bilgilerini saklar.
mapred-site.xml	JobTracker'in kullanacağı host ve port numaralarını saklar.
hdfs-site.xml	HDFS dosya sistemi ile ilgili ayarları saklar. Her bir dosyanın yedeklenme sayısı vb. parametreler buraya yazılmalıdır.

Kurulumu yapılmış tek node'luk Hadoop sistemi için yapılandırma dosya ayarlarının ekran görüntüsü Şekil 6.2' deki gibidir. Hdfs-site.xml dosyasının içindeki dfs.reliplication parametresiyle Hadoop sistemi üzerinde saklanan verilerin kaç yedeğinin oluşturulacağı belirlenmektedir. Kurulumu yapılan tek node'luk Hadoop kümesi için bu değer 1 iken 4 node'luk Hadoop kümesi için bu parametre 3 olarak belirlenmiştir.

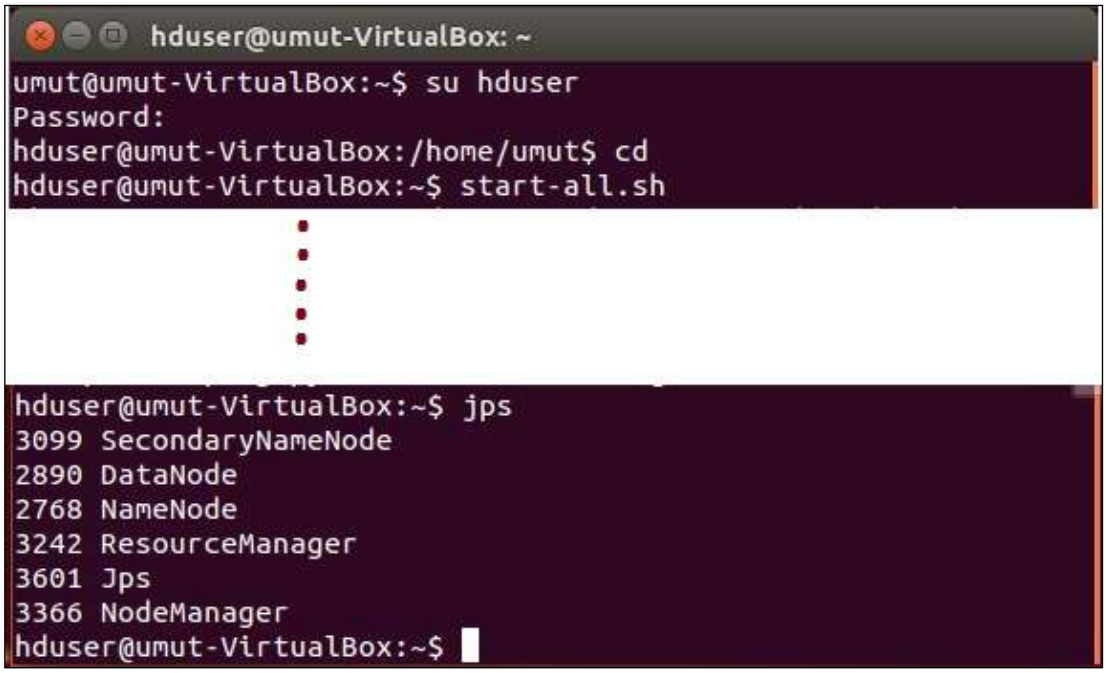


Şekil 6.2: Tek node'luk hadoop kurulum ayarları

Hadoop kurulumunda yukarıda belirtilen ayarlamalar yapıldıktan sonra;

- Komut satırından NameNode formatlanır. NameNode formatlanması, HDFS dosya sisteminin kullanılması ve başlatılması için gereklidir. Formatlama işlemi "*hadoop namenode –format"* komutu ile gerçekleştirilir.
- Bu aşamada Hadoop kurulumu gerçekleştirilmiştir. Hadoop'u çalıştırmak için komut satırından Hadoop dosyalarının bulunduğu dizinine gidilerek *"start-all.sh"* komutu yazılırsa Hadoop sistemi çalışmaya başlayacaktır. Hadoop'u durdurmak için benzer şekilde ilgili dizine gidilir ve "*stop-all.sh"* komutu kullanılır.

Hadoop çalıştırıldıktan sonra *jps* komutu ile çalışan Hadoop işleçleri görüntülenebilir. Şekil 6.3'de tek node'luk bilgisayarda Hadoop'un komut satırından çalıştırılmış halinin ekran görüntüsü mevcuttur. Jps komutu ile çalışan Hadoop bileşenleri listelenmiştir.

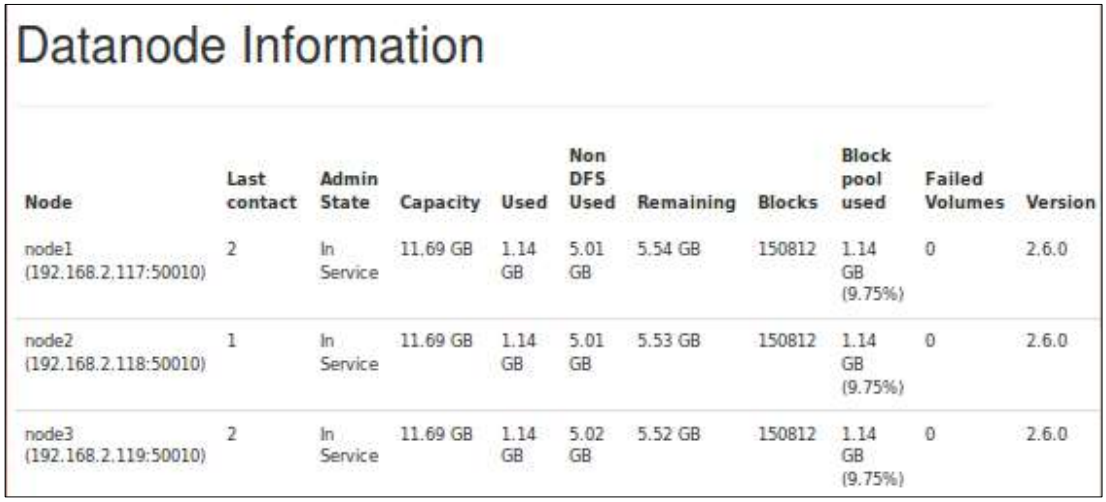


Şekil 6.3: Tek node'luk hadoop'un çalıştırılması

Hadoop geliştiricileri kurulum için 3 farklı model önermektedir. Bu kurulum modelleri Standalone Mod, Pseudo-Distributed Mod ve Fully Distributed Mod şeklindedir. Tek node'luk için yapılan kurulum standalone mod kurulumudur. Pseudo-distributed mod kurulumu diğer iki kurulum şeklinin arasında bir kurulum yöntemidir. Bu yöntem bir node üzerinde Hadoop işlerinin dağıtık java görevleri olarak çalıştırılmasıdır. Fully distributed mod ise Hadoop'un bir küme üzerinde çalıştırılmasıdır(Hadoop mode 2016). Yapılan 4 node'luk Hadoop kurulumu fully distributed mod yapısındadır.

Çalışma kapsamında kullanılacak olan Hadoop küme kurulumu ayarları tek node'luk kurulum ayarlarından çok farklı değildir. Her bir bilgisayar için sabit bir IP(Internet Protocol Address) adresi belirlenmeli ve Hadoop kullanıcıları arasında SSH anahtarının paylaşılması gereklidir. Hadoop dizininde bulunan conf klasörünün içerisindeki master ve slave dosyalarının içerisine master ve slave olan bilgisayarların bilgileri(Ip adresleri ve isimleri) yazılmalıdır. Hadoop'un çalıştırılması ve durdurulması tek node'luk kurulumda kullanılan komutlarla yapılabilir. Bu tez çalışması için oluşturulan 4 node'luk Hadoop kurulumu için öncelikle tek node'luk kurulum gerçekleştirilmiş daha sonra sanal bilgisayar programı olan virtualbox yardımıyla sanal bilgisayarın 3 ayrı kopyası oluşturulmuş, isimleri ve diğer ayarları değiştirilerek 4 node'luk Hadoop kümesi oluşturulmuştur.

Hadoop HDFS dosya sistemi ve görevlerin takibi için web arayüzü sağlamaktadır. HDFS dosya sistemi için sunmuş olduğu ara yüz yardımıyla küme içerisindeki bilgisayarların çalışma durumları, her bilgisayardaki veri blok sayıları ve kullanılan depolama alanı gibi bilgileri sunmaktadır. Şekil 6.4'te 4 node'luk kümedeki bilgisayarların hdfs bilgileri görülmektedir. Burada her node üzerinde saklanan veri bloklarının sayıları, kullanılan hafıza boyutu vb. bilgiler gözlemlenebilmektedir.

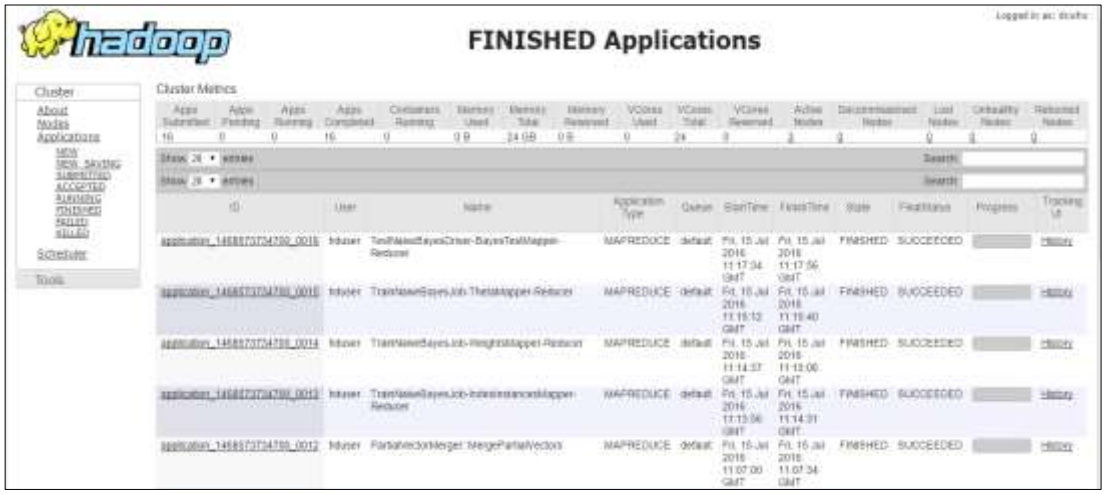


Datanode Information

Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
node1 (192.168.2.117:50010)	2	In Service	11.69 GB	1.14 GB	5.01 GB	5.54 GB	150812	1.14 GB (9.75%)	0	2.6.0
node2 (192.168.2.118:50010)	1	In Service	11.69 GB	1.14 GB	5.01 GB	5.53 GB	150812	1.14 GB (9.75%)	0	2.6.0
node3 (192.168.2.119:50010)	2	In Service	11.69 GB	1.14 GB	5.02 GB	5.52 GB	150812	1.14 GB (9.75%)	0	2.6.0

Şekil 6.4: Hdfs node bilgileri

Hadoop çalıştırılan uygulamalarla ilgili bilgilerin takibi için de web arayüzü sağlamaktadır. Şekil 6.5'te 4 node'luk hadoop kümesi üzerinde sınıflandırma işlemi sırasında oluşturulan görevler ve görevlere ait bilgiler bulunmaktadır.



Şekil 6.5: Hadoop üzerinde çalıştırılan görevler

Her göreve ait başlama bitiş süreleri, görevin durumu, görevin adı, görevin yaptığı işlere ait log bilgileri vb. bilgilere bu arayüz yardımıyla erişilebilmektedir.

Benzer şekilde arayüz yardımıyla yeni bir görev oluşturulabilir ve küme üzerinde çalıştırılabilmektedir. Yine bu arayüz yardımıyla kümede bulunan node'lara ait bilgilere de erişilebilmektedir.

6.1.1 Mahout Kurulumu

Tez çalışması kapsamında veri madenciliği için mahout aracı kullanılacaktır. Mahout Hadoop üzerinde çalışan bir java kütüphanesidir. Ubuntu işletim sisteminde mahout kurulumu için geliştirilmiş bir araca rastlanmamıştır. Bu nedenle komut satırından adım adım kurulumu gerçekleştirilmiştir. Kullanılan kurulum yönteminde mahout'un kurulumu için java ve hadoop'un yanı sıra maven'in kurulu olması gerekmektedir(Mahout Kurulum 2016). Bu nedenle Maven(versiyon 3.3.3) kurulumu gerçekleştirilmiştir. Hadoop kümesi için bu kurulum sadece master bilgisayar üzerinde gerçekleştirilmiştir.

Maven kurulumundan sonra mahout(versiyon 0.11.0) kurulumu komut satırından gerçekleştirilmiştir. Mahout kurulumunun tamamlandığını test etmek için komut satırına *"mahout"* komutu yazılır, ekranda mahout için kullanabilecek parametrelerin listesi geldiğinde mahout kurulumu tamamlanmış olacaktır.

Şekil 6.6: Bilgisayarlarda kurulan yazılımların mimarisi

Şekil 6.6'da 4 node'luk küme içerisindeki master bilgisayar ve tek node'luk bilgisayarda kurulumu yapılmış olan kütüphanelerin/uygulamaların mimari olarak görseli mevcuttur. Mahout kurulumundan sonra tez çalışmasında yapılacak olan veri madenciliği için altyapı hazır hale gelmiş bulunmaktadır.

6.2 Veri Kümesinin Oluşturulması

Veri madenciliği için kullanacak olan veri kümesi Türkiye'deki günlük gazetelerin Twitter'da paylaşmış oldukları haber başlıkları olarak belirlenmiştir. İnternet kullanımın artması ve insanların her yerden çeşitli cihazlarla internete erişim sağlaması sonucu gazetelerin web ortamında okunma seviyesi artmıştır. Gazeteler gün içindeki gelişmelere dair web sitelerinde haber yapmaktadırlar. Yapmış oldukları haberleri yine kendilerine ait olan sosyal medya hesapları üzerinden paylaşmaktadırlar. Paylaşım yaptıkları sosyal medya hesaplarına Facebook, Twitter, Instagram, Google+ ve Youtube örnek olarak verilebilir.

Sosyal medya üzerinden yapılan paylaşımların nedenleri arasında daha fazla kitleye ulaşmak, günlük olarak yaşanan gelişmelerden insanları haberdar etmek vardır. Sosyal medya üzerinden paylaşılan bir haber için ilgili okuyucu haber linkine tıklayarak haberin tam metnini görmek isteyebilir. Okuyucuların linklere tıklanması durumunda ilgili gazeteye reklam gelirlerinden pozitif değer sağlamış olmaktadır. Bu nedenle haber başlıkları genellikle insanların ilgilerini çeken, merak uyandıran ifadelerle verilmektedir. Ayrıca haber başlıkları haber ile ilgili detayı genellikle vermemektedir. Yani sadece haber başlığını okuyarak konuyla ilgili olarak bir yargıya varmak çok kolay değildir.

Bu tez çalışmasında veri madenciliği yöntemleri kullanılarak haber başlıkları üzerinde duygu analizi yapılacaktır. Haber başlıkları olumlu veya olumsuz olarak sınıflandırılacaktır. Duygu analizi için kullanılacak olan haber başlıkları duygu analizi için zor bir veri kümesidir. Çünkü haber başlıklarının ilk okuyuşta olumlu veya olumsuz olduğuna karar vermek zordur. Bundan dolayı çok uç noktalarda olmamak kaydı ile birçok haber başlığı genellikle belirsiz bir anlam içermektedir. Bu belirsizlik gazeteler için bir başarı kriteri olarak görülebilir. Bu belirsizliği gidermek için çoğu

zaman okuyucu haberin linkine tıklayarak haberin içeriğini anlamak istemektedir. Bu tıklama istekleri gazeteler açısından istenilen bir okuyucu davranışıdır.

Gazeteler Twitter'da paylaşmış oldukları mesajların içeriği genellikle "haber başlığı + haber linki", "fotoğraf + haber başlığı + haber linki", "fotoğraf + haber başlığı" veya "fotoğraf + haber linki" şeklindedir. Twitter API'sinden alınan bu formattaki tweet mesajları, bazı ön işlemlerden geçirilmiştir. Ön işlemden sonra sadece haberin başlığının içeriği metin dosyası olarak kaydedilmiştir. Ön işlemden önce bir tweet'e ait, atılma tarihi, Twitter kullanıcı adı, kullanıcının görünen tam ismi ve atılan tweet cümlesi API'den alınmıştır.

6.2.1 Twitter API'si ve Twitter4j Kütüphanesi

Veri kümesi için sosyal medya araçlarından Twitter tercih edilmiştir. Twitter'in günümüzde çok yaygın bir kullanım oranı mevcuttur. Büyük veri bölümünde Twitter kullanımına dair bazı istatistikler paylaşılmıştı. Paylaşılan değerler göz önüne alındığında sosyal medyada üretilen verilerin bir büyük veri problemi olduğu aşikârdır. Bu büyük veri kümesini oluşturan sosyal paylaşım sitelerinden Twitter aracının tercih edilmesindeki bazı faktörler aşağıdaki gibidir;

- Yaygın kullanım oranı
- Erişim kolaylığı
- Çok güçlü ve yetenekli API'si
- API'nin ücretsiz kullanılması
- API'nin birçok bilgiyi paylaşması

Bu avantajların yanında API'nin kullanım kısıtlamaları da mevcuttur. Bu kısıtlamalardan bazıları; bir kullanıcıya bir seferde en fazla 3200'e yakın tweet vermesi, geçmişe yönelik belli bir tarihten önceki tweet'lerin sorgu sonucunda getirilmemesi örnek verilebilir.

Twitter API'sinden gazetelerin haber başlıklarının alınması için Twitter4j açık kaynak kodlu kütüphanesi kullanılmıştır(Twitter4j 2016). Twitter4j API ile kolay bir şekilde haberleşebilmektedir. Sağlamış olduğu fonksiyonlar yardımıyla API'den

istenilen veriler çekilebilmektedir. Twitter API'sinden Twitter4j kütüphanesi yardımıyla veri çekebilmek için bir Twitter hesabı açtıktan sonra bir twitter uygulaması oluşturulmalıdır. Oluşturulan twitter uygulamasının vermiş olduğu Consumer Key (API Key), Consumer Secret (API Secret), Access Token ve Access Token Secret anahtar değerleri Twitter4j kütüphanesi için kullanılmıştır. Bu şekilde Twitter API'sinden veriler alınmıştır.

6.2.2 Verilerin API'den Çekilmesi

Twitter'dan verilerin alınması için java diliyle bir masaüstü uygulaması geliştirilmiştir. Uygulama her bir gazete için tek tek çalıştırılmıştır. Uygulamanın her çalıştırılmasında her gazete için daha önce atılmış olan 3200 civarında tweet alınmış ve cümleler ön işlemlerden geçirilerek metin formatında kayıt edilmiştir. Ön işlemden önce alınan bir tweet cümlesi aşağıdaki gibidir. Önişlemlerde tarih ve diğer kullanıcı hesaplarına ait bilgiler silinmiştir. Geriye kalan tweet cümlesi, atılmış olduğu tarih adıyla bir metin olarak kaydedilmiştir.

Mon Feb 22 22:33:17 EET 2016 Aksam Aksam.com.tr DİKKAT! Emniyet Genel Müdürlüğü 4 bin memur alacakhttps://t.co/IrKKlod48X https://t.co/UAtytqZRYn

Yukarıdaki tweet cümlesi önişlem aşamasından sonra aşağıdaki şekildeki haliyle kaydedilmiştir.

DİKKAT! Emniyet Genel Müdürlüğü 4 bin memur alacakhttps://t.co/IrKKlod48X https://t.co/UAtytqZRYn

API'nin kısıtlaması nedeniyle bir seferde 3200'den fazla tweet alınamamıştır. Bunun yanında API, bir uygulamadan gelen bir istekten sonra genellikle 15 dakika boyunca aynı uygulamadan gelen yeni bir isteğe cevap vermemektedir. Bu nedenle yazılan uygulama 20 dakika da bir periyodik olarak yeni bir gazetenin hesabındaki mesajları almak için otomatikman çalıştırılacak şekilde geliştirilmiştir. Uygulama 4 farklı tarihte çalıştırılmıştır. Uygulama çalıştırma zamanları Tablo 6.2'de yazılmıştır.

Tablo 6.2: Veri çekme uygulamasının çalışma zamanları

Uygulama Çalıştırma Sırası	Çalıştırma Zamanı
1. Çalıştırma	22 Şubat 2016
2. Çalıştırma	10 Mart 2016
3. Çalıştırma	07 Nisan 2016
4. Çalıştırma	15 Nisan 2016

Her uygulama çalıştırma işleminden sonra alınan tweet cümleleri daha önce alınan tweet cümleleriyle elle birleştirilerek tek dizinde saklanmaktadır. Birleştirme işleminde API'ye gönderilen her istekte, API son 3200 civarında tweet cümlesi geri döndürmektedir. Bu nedenle uygulamanın iki kez(1. Çalıştırma ve 2. Çalıştırma gibi) çalıştırılması sonucunda API aynı tweet cümlesini geri gönderebilmektedir. Her çalıştırmada alınan tweet cümleleri alınan tarih ve saat bilgisiyle kayıt edildiğinden, iki farklı çalıştırma sonucunda elde edilen tweet'ler tarihler referans alınarak birleştirilmiştir. Bu şekilde aynı tweet cümlesinin iki veya daha fazla kez veri kümesine eklenmesi engellenmiş olmaktadır.

Tablo 6.3'te uygulamanın çalıştırılması sonucu her gazeteye ait alınan tweet kümelerindeki tweet mesajlarından ilk alınan ve son alınanın mesajların tarihleri ve alınan toplam tweet sayıları gösterilmiştir. Bu tabloda yazılan tweet sayıları eğitim verisi oluşturma aşamasından önceki rakamlardır. Eğitim verisi oluşturma aşamasında bazı mesajlar silinmiştir. Silinen mesajların nedenleri eğitim verisi hazırlama bölümünde detaylıca açıklanacaktır. Her gazeteye(bugün gazetesi hariç) ait bitiş tarihlerinin aynı olmasının nedeni, tweet alma uygulamasının en son 15 Nisan 2016 tarihinde çalıştırılmış olmasıdır. Bugün gazetesinin resmi Twitter hesabında 30 Kasım 2015 tarihinden sonra bir paylaşım yapılmamıştır. Benzer şekilde her gazetenin başlangıç tarihlerinin farklı olmasının nedeni ise uygulamanın birinci çalıştırılması sonucunda alınan 3200'e yakın tweet arasında atılmış olan en eski tweet'in tarihinin olmasıdır. Her gazeteye ait başlangıç ve bitiş tarihleri arasında atılan tüm tweet'lerin alınması hedeflenmiştir. Twitter API'si veya uygulama taraflı oluşmuş olan hatalardan dolayı tam olarak iki tarih aralığındaki tüm tweet'ler alınmamış olabilir.

Tablo 6.3: Alınan tweet'lerin tarihleri ve sayıları

Gazete Adı	Başlangıç Tarihi	Bitiş Tarihi	Alınan Tweet Sayısı
Akşam	23 Ocak 2016	15 Nisan 2016	8581
Bugün	20 Ekim 2015	30 Kasım 2015	2640
Fanatik	29 Ocak 2016	15 Nisan 2016	7922
Güneş	23 Kasım 2015	15 Nisan 2016	4998
Habertürk	01 Şubat 2016	15 Nisan 2016	14594
Hürriyet	21 Ocak 2016	15 Nisan 2016	9526
Milliyet	29 Ocak 2016	15 Nisan 2016	8981
Posta	22 Ocak 2016	15 Nisan 2016	8230
Sabah	27 Ocak 2016	15 Nisan 2016	7233
Sözcü	17 Şubat 2016	15 Nisan 2016	6164
Star	06 Şubat 2016	15 Nisan 2016	9250
Takvim	03 Aralık 2015	15 Nisan 2016	4938
Türkiye	03 Aralık 2015	15 Nisan 2016	4667
Vatan	14 Ocak 2016	15 Nisan 2016	4435
Yenişafak	21 Ocak 2016	15 Nisan 2016	2372
		Toplam	**104531**

6.2.3 Eğitim Verisinin Oluşturulması

Gazetelerin resmi hesaplarından alınan tweet cümleleri metin olarak kaydedildikten sonra bu verilerin sınıflandırma algoritması için hazırlanması gerekmektedir. Sınıflandırma algoritmalarından Naive Bayes için eğitim verisi oluşturmak amacıyla bir java masaüstü uygulaması geliştirilmiştir. Sınıflandırma algoritması için haber başlıkları olumlu veya olumsuz olarak belirlenmek istenmektedir. Yani sınıflandırma algoritması için iki sınıf olacaktır, bunlar olumlu ve olumsuz sınıflarıdır. Bu şekilde bir haber başlığı için duygu analizi yapılmış olacaktır. Haber başlıklarındaki duygular 3 sınıfa ayırılmıştır, olumlu veya olumsuz olarak işaretlenemeyen haber başlıkları belirsiz olarak işaretlenmiştir. Belirsiz sınıfı tez çalışması kapsamında kullanılmayacaktır. Belirsiz sınıfındaki haber başlıkları genellikle büyük anlam belirsizliği içermektedir.

Bir metin üzerinde duygu analizi yapmak için dilin yapısı, kelimelerinin yapısı veya dilin biçimi gibi ölçütler söz konusudur. Bu nedenle duygu analizi için metindeki kelimeleri tek tek işlenmesi gerekmektedir. Bu işlemler için doğal dil işleme yöntemleri kullanılmaktadır. Doğal dil işleme bir dili dijital ortamda inceleme yöntemleridir. Bir metindeki duygunun ortaya çıkarılması, bilgisayar sitemleri tarafından insanlar gibi algılanması için doğal dil işleme tekniklerine ihtiyacı doğurmuştur. Bu tez çalışmasında Türkçe doğal dil işleme işlemleri için geliştirilmiş olan Zemberek(Zemberek 2016) kütüphanesi kullanılmıştır. Zemberek kütüphanesi Türkçe doğal dil işleme için önemli bir kütüphanedir. Bu kütüphane yardımıyla, bir kelimenin ekleri, heceleri, tipi gibi özellikler belirlenebilmektedir. Eğitim verisi oluşturmak için yazılan uygulama zemberek işlemelerinden önce bir cümle üzerinde aşağıdaki işleri yapmaktadır.

- Cümle içindeki linkleri temizleme
- Kesme işaretinden sonraki kısımların temizlenmesi(örneğin 5'inci kelimesindeki -'inci- kısmının temizlenmesi)
- Eğer cümle Türkçe haricinde bir dilde yazılmışsa tamamen silinmesi
- Retweet edilen cümlelerdeki "RT @aksamspor:" alanları temizlenmesi
- Karakter haricinde yazılan sembollerin temizlenmesi(örneğin ⮷, □, ▸ gibi)
- Bazı birimlerin temizlenmesi(örneğin, kg, cm, m gibi)
- Bütün noktalama işaretlerinin temizlenmesi
- #, &, $, @ gibi bütün sembollerin temizlenmesi
- Bütün cümlenin küçük harflere dönüştürülmesi

Zemberek kütüphanesi ile her tweet cümlesi için aşağıdaki işlemler gerçekleştirilmiştir.

- Cümledeki edat ve bağlaçların temizlenmesi
- Kelimenin doğru yazılışının denetlenmesi
- Yanlış yazılan kelimeler için kelime önerilmesi
- Çekim eklerini belirlenmesi ve silinmesi
- Yapım eklerinin belirlenmesi

Bu temel adımlardan sonra zemberek yardımıyla kelimenin kökü ve ekleri değerlendirilmiştir. Zemberek kütüphanesi bir kelimeye ait aşağıdaki gibi birden fazla çözümleme önerisinde bulunmaktadır. Bu çözümlemelerden birincisi kabul edilmiş ve gövdeleme işlemi onun üzerinden yapılmıştır. Örneğin; ”Yanındaki” kelimesine ait zemberek tarafından 6 adet çözüm önerilmiştir. Bu çözümler Tablo 6.4’teki gibidir. Bir cümlede yanındaki kelimesinin zemberek kütüphanesine göre çözümlenmesi sonucu 6 farklı kullanımı mevcuttur.

Tablo 6.4: "Yanındaki" kelimesine ait çözümlemeler

	Kelime	**Çözümleme**
Çözüm-1	Yanındaki	[Kok: yan, ISIM] Ekler: ISIM_TAMLAMA_IN + ISIM_KALMA_DE + ISIM_BULUNMA_KI
Çözüm-2	Yanındaki	[Kok: yan, ISIM] Ekler: ISIM_TAMLAMA_I + ISIM_KALMA_DE + ISIM_BULUNMA_KI
Çözüm-3	Yanındaki	[Kok: yan, ISIM] Ekler: ISIM_SAHIPLIK_O_I + ISIM_KALMA_DE + ISIM_BULUNMA_KI
Çözüm-4	Yanındaki	[Kok: yan, ISIM] Ekler: ISIM_SAHIPLIK_SEN_IN + ISIM_KALMA_DE + ISIM_BULUNMA_KI
Çözüm-5	Yanındaki	[Kok: ya, ISIM] Ekler: ISIM_TAMLAMA_IN + ISIM_KALMA_DE + ISIM_BULUNMA_KI
Çözüm-6	Yanındaki	[Kok: ya, ISIM] Ekler: ISIM_SAHIPLIK_SEN_IN + ISIM_TAMLAMA_IN + ISIM_KALMA_DE + ISIM_BULUNMA_KI

Kelimedeki çekim ekleri(ler, lar, yor vb.) silinerek her kelime için gövdeleme işlemi yapılmıştır. Gövdeleme işleminde silinen ekler yerine ek önerme fonksiyonu kullanılmıştır. Yukarıda bahsedilen bütün adımlar eğitim verisi oluşturma uygulamasıyla yapılmıştır.

Şekil 6.7’de eğitim verisi oluşturmak için yazılmış olan masaüstü uygulamasının ekran görüntüsü mevcuttur. Uygulama yardımıyla metin formatında kayıt edilen tweet’ler sınıflara ayırılmak istenmiştir.

Şekil 6.7: Eğitim verisi oluşturma uygulaması

Uygulama java programla diliyle geliştirilmiştir. Uygulamanın çalışması şöyledir; hangi gazeteye ait tweet'ler sınıflandırılacaksa dosya okuma yoluna o gazete dosyalarının bulunduğu dizininin adresi yazılır. Olumlu, olumsuz veya belirsiz olarak işaretlenecek mesajların kayıt edileceği dosya yolları da ilgili alanlara yazılır. Başla butonuna tıklandıktan sonra "Tweet Cümlesi" alanına gazete tarafından atılan haber başlığı gelir. Bu işlemde dizinde bulunan metin dosyalarından birincisinin içeriği ekrana gelir. Okunan haber başlığının sınıfını belirlemek için olumlu, olumsuz veya belirsiz butonuna tıklanır. Okunan tweet cümlesinin olumlu, olumsuz veya belirsiz olmasına karar verildiğinde, olumlu, nötr veya olumsuz butonlarından birine tıklanır. Bu tıklama işleminden sonra cümle üzerinde yukarıda bahsedilen doğal dil işleme adımlar uygulanır ve verilen bir id adıyla ilgili dizine kayıt edilir. Tweet'leri saklayan her bir metin dosyasının adı bir tam sayı ile değiştirilmiştir. Dosya isimleri 1 den başlatılmış yaklaşık 105000'e kadar gitmiştir.

Uygulamadaki sonraki butonun görevi ise, ekranda beliren cümlenin içeriği uygun olmadığında işaretlemeden bir sonraki cümleye geçmeyi sağlamaktır. Bu şekilde farklı dillerde atılan tweet'ler geçilmiş ve veri kümesine alınmamıştır. Gazeteler genellikle gün içinde aynı haberi hiçbir değişiklik yapmadan günün farklı saatlerinde paylaşmaktadırlar, bu haberler başlıkları için sonraki butonu kullanılmıştır.

Benzer şekilde gazeteler bazen bir fotoğraf üzerine yazılmış bir haber başlığı ve bir link paylaşmışlardır. Bir metin olmadığından bu tweet'ler içinde sonraki butonu kullanılmıştır.

Bitir butonu ise, başta uygulamanın kapatılması görevini gerçekleştirir. Diğer görevleri arasında uygulamayı kapatmadan önce, uygulamanın bir sonraki çalıştırılmasında ihtiyaç duyulacak en son kullanılan id değeri, gazetenin son okunan tweet'inin adı vb. bilgileri metin dosyası olarak kaydetmektir.

Her haber başlığı için gövdeleme işlemi gerçekleştirilmiştir. Örneğin; bir cümle gövdeleme işleminden önce "ilkokul mezunu şadi başer otomobillerde yüzde 40 yakıt tasarrufu sağlayan cihaz geliştirdi" şeklindedir. Gövdeleme işleminden sonra bu cümle "ilkokul mezun şadi başer otomobil yüzde 40 yakıt tasarruf sağla cihaz geliştir" şeklini almıştır. Eğitim verisi oluşturma aşamasından sonra her gazete ait olumlu, olumsuz veya belirsiz tweet'lerin sayısı Tablo 6.5'teki gibidir.

Tablo 6.5 incelendiğinde belirsiz haberlerin sayısı diğer sınıflara göre daha fazladır. En az tweet sayısı olumlu haber sınıfındadır. Gazetelerin Twitter uygulamasını kullanım oranlarında ise en çok tweet atan habertürk gazetesi, en az tweet atan gazete ise 4435 sayısıyla vatan gazetesi olmuştur. Yenişafak gazetesinin yaklaşık 9 bin civarında tweet cümlesi Twitter'dan alınmıştır bunun tablodaki kısmı kadarı eğitim verisi olarak hazırlanabilmiştir. Bugün gazetesinin ise Twitter hesabında güncel olarak çok fazla tweet paylaşılmamıştır. Bu iki gazete bahsedilen nedenlerden dolayı ölçeklendirmeye dâhil edilmezse, en az tweet'i vatan gazetesi atmış olacaktır. Bu değerler daha önce de belirtilen tarihler arasındaki gazetelerin tweet paylaşımlarından oluşmaktadır.

Tablo 6.5'te görüldüğü üzere olumlu ve olumsuz sınıftaki toplam tweet sayıları farklıdır. Bu şekilde sınıflandırma algoritması cümle sayısı fazla olan sınıftan yana karar verme eğilimine girebilir. Bu eğitim verisi sınıflardaki tweet sayıları arasında çok fark olduğundan ideal bir eğitim verisi değildir. Yapılacak uygulamalarda eğitim ve test verisindeki tweet sayıları rastgele seçimlerle eşit veya yakın sayılarda olacak şekilde ayarlanacaktır. Eğitim verisi çeşitli oranlarda bölünerek uygulamalar yapılacaktır.

Tablo 6.5: Gazetelerin olumlu, olumsuz ve belirsiz başlık sayıları

Gazete Adı	**Olumlu Sayısı**	**Belirsiz Sayısı**	**Olumsuz Sayısı**	**Toplam**
Akşam	924	5552	2105	8581
Bugün	323	1112	1205	2640
Fanatik	1708	4230	1984	7922
Güneş	337	3334	1327	4998
Habertürk	2122	8853	3619	14594
Hürriyet	1078	5643	2805	9526
Milliyet	1378	4943	2660	8981
Posta	1235	3388	3607	8230
Sabah	693	4612	1928	7233
Sözcü	1664	1217	3283	6164
Star	966	4662	3622	9250
Takvim	1518	1471	1949	4938
Türkiye	1471	1409	1787	4667
Vatan	515	2860	1060	4435
Yenişafak	477	994	901	2372
Toplam	16409	54280	33842	102159

6.3 Çalışma ve Sonuçlar

Hazırlanan çalışma ortamı ve oluşturulan eğitim verisinden sonra iki ayrı Hadoop ortamı için veri kümesinin belli kullanım şekilleriyle sınıflandırma işlemleri gerçekleştirilmiştir. Yapılan her çalışma ayrı bir uygulama başlığı altında anlatılmıştır.

Mahout sınıflandırma algoritmaları için çeşitli parametreler sunmaktadır. Yapılan uygulamalarda bu parametrelerden bazıları değiştirilerek deneyler yapılmış ve sonuçları yazılmıştır. Mahout'un sunmuş olduğu bu parametreler arasından en önemlilerinden bir tanesi n-gram modelidir. N-gram modeli, karakterlerden oluşan bir dizgenin belli bir parçasıdır (Meral ve Diri 2014). Örneğin, "Merhaba nasılsınız" cümlesinin 2-gramları ve 3-gramları aşağıdaki gibidir.

- 2-gramlar: ”me”, ”er”, “rh”,”ha”, “ab”, “ba”, “a ”, “ n”, “na”, “as”, “sı”, “ıl”, “ls”, “sı”, “ın”, “nı”, “ız”
- 3-gramlar: ”mer”, ”erh”, ”rha”, ”hab”, ”aba”, ”ba ”, ”a n”, ” na”, ”ası”, ”sıl”, ”ıls”, ”lsı”, ”sın”, ”ını”, ”nız”

Mahout ile metin içerisindeki kelimeleri n-gram parametresiyle istenilen boyutta alt parçalara bölünebilir. Bu şekilde her dokümana ait karakter dizileri oluşturularak her doküman özellik vektörleriyle ifade edilecektir. Mahout üzerinde Naive Bayes algoritmasıyla sınıflandırma yapmak için kullanılan temel komutlar ve parametreler aşağıdaki gibidir(Mahout Parametreler 2016).

- Seqdirectory: Bu komut sınıflandırma için dizinde bulunan dosyaları Mahout’un kullandığı SequenceFile formatına dönüştürür. Komutun yanında giriş dizinin adresi, çıktı dizin adresi gibi parametreler yazılır.
- Seq2sparse: SequenceFile dosyasından özellik vektörü oluşturur. Bu komutun önemli parametreleri arasında özellik ağırlıklandırma yöntemi olan tf-idf vardır. Bunun yanında kelimelerin hangi n-gram modeline göre ayrılacağı da bu komuta parametre olarak yazılmaktadır.
- Trainnb: Algoritmanın eğitilmesini sağlamaktadır. Parametre olarak eğitim kümesi vektörünü alır, çıktı olarak eğitilmiş model üretir.
- Testnb: Algoritmanın sınıflandırma başarımı belirlemek için kullanılır. Giriş parametreleri olarak modelin dosya yolu ve test vektörünü alır. Çıkış parametresi olarak da model test sonuçları için çıktı dosya yolunu alır. Bu komutun çalıştırılmasından sonra sınıflandırma modelinin başarım ölçütleri konsol ekrana yazılmaktadır.

6.3.1 Uygulama-1

Bu uygulamada tek node’luk hadoop ve mahout sistemi üzerinde farklı kategoriler de haber yapan gazetelere ait tweet’ler kategorilerine göre sınıflandırılmıştır. Sınıflandırmadaki amaç bir haberin hangi gazeteye ait olduğuna bilgisayarın karar verebilmesidir. Sınıflandırmada kullanılmak üzere Posta Gazetesi, Fanatik Gazetesi ve Yenişafak Gazetesi’ne ait veriler seçilmiştir. Bu gazetelere ait olan

tweet cümlelerinden bir kısmı eğitim bir kısmı ise test verisi olarak kullanılmıştır. Bu uygulamadaki amaç yeni gelen bir haber başlığının hangi gazeteye ait olduğuna karar verebilmek için bir sınıflandırma modeli oluşturmaktır. Oluşturulan sınıflandırma modelinin başarısını test etmektir. Mahout bir metindeki kelimeleri özellik olarak belirlemek için g-gram modelini kullanmayı kullanıcılara sunmaktadır.

Uygulama için dengeli ve dengesiz olan veri kümeleri 2-gram ve 3-gram yöntemlerini kullanılarak sınıflandırılmıştır. Öncelikle dengesiz veri kümesi 2-gram modeline göre 5 defa çalıştırılmış başarım sonuçları kaydedilmiş, daha sonra aynı veri kümesi 3-gram modeline göre de 5 defa çalıştırılmış ve sonuçlar kaydedilmiştir. Benzer şekilde aynı işlemler dengeli veri kümesi için de yapılmıştır.

Tablo 6.6'da dengesiz eğitim ve test veri kümelerindeki tweet cümlelerinin sayıları görülmektedir. Dengesiz eğitim verisinde, her gazeteye ait bütün tweet'lerin %66'sı eğitim, kalan kısmı ise test verisi olarak ayırılmıştır. Bu ayırma işlemi yazılan bir uygulama yardımıyla rastgele metin dosyalarının seçilmesiyle yapılmıştır.

Tablo 6.6: Dengesiz eğitim ve test veri kümesi sayıları

	Eğitim Verisi(%66)	**Test Verisi(%33)**
Posta	3050	1572
Yenişafak	1564	806
Fanatik	5223	2691

Tablo 6.7'de ise dengeli eğitim verisinin tweet sayıları yazılmaktadır. Metin dosyalarının sayılarının aynı olması için üç gazeteden en az sayıya sahip gazetenin metin dosyalarının sayıları kadar diğer iki gazetede ayarlanmıştır.

Tablo 6.7: Dengeli eğitim ve test veri kümesi sayıları

	Eğitim Verisi(%66)	**Test Verisi(%33)**
Posta	1564	806
Yenişafak	1564	806
Fanatik	1564	806

Her iki veri kümesi de Hadoop üzerinde çalıştırılmıştır. Sınıflandırma sonuçları Tablo 6.8'de gösterildiği gibidir. Sınıflandırma işlemindeki doğruluk değeri; doğru

olarak sınıflandırılmış tweet metinlerinin, tüm metin sayısına oranıdır. Sınıflandırmada kullanılan bir diğer kriter olan hata oranı ise doğruluk değerinin birden çıkarılması sonucu bulunan değerdir. En yüksek doğruluk değeri, dengesiz veri setinin 2-gram modeline göre sınıflandırılmış veri kümesinde elde edilmiştir. En düşük doğruluk değeri ise 3-gram modeli ve dengeli veri kümesinde elde edilmiştir. Burada her iki veri kümesi içinde 2-gram yöntemi 3-gram yöntemine göre daha başarılı sonuçlar üretmiştir.

Tablo 6.8: Uygulama-1 sınıflandırma sonuçları

	N-gram	**Doğruluk(%)**	**F1 Değeri**
Dengeli	2-gram	76.139	0.7589
Dengeli	3-gram	75.384	0.7518
Dengesiz	2-gram	79.847	0.8004
Dengesiz	3-gram	78.796	0.7915

Sınıflandırma başarım değerlerinden biri olan F1 değeri, hassasiyet(Precision) ve getirim(Recall) ile ilgilidir. Bu iki değerin harmonik ortalamasından oluşmaktadır. F1 değerinin alabileceği en büyük değer 1'dir. F1 değeri ne kadar yüksekse yapılan sınıflandırma başarımı o kadar iyidir. Tablo 6.8'de de görüldüğü üzere doğruluk değeri ile F1 değeri arasında doğru orantı vardır. F1 değeri bir sonraki uygulamada detaylıca açıklanmıştır.

6.3.1 Uygulama-2

Bu uygulamada tek node'luk ve 4 node'luk Hadoop kümeleri üzerinde haber başlıkları olumlu ve olumsuz olarak sınıflandırılacaktır. Uygulama-1'deki sonuçlar dikkate alındığında veri kümesi için 2-gram modeli daha başarılı sonuçlar vermiştir. Bu nedenle bu uygulamada 2-gram yöntemi tercih edilmiştir. Bu uygulama için eğitim kümesi dengesiz bir dağılıma sahiptir. Eğitim ve test verisi arasında yaklaşık olarak % 66 eğitim yaklaşık % 33 test verisi olarak belirlenmiştir. Eğitim ve test verilerin belirlenmesi yazılan bir uygulama tarafından sağlanmıştır. Uygulama rastgele metin dosyalarının % 66'sını eğitim verisi olarak ayırmıştır. Eğitim ve test veri kümelerindeki tweet sayılarının (metin dosyası sayılarının) dağılımı Tablo 6.9'da yazıldığı gibidir.

Tablo 6.9: Uygulama-2 eğitim ve test verisi tweet sayıları

	Eğitim Verisi(%66)	**Test Verisi(%33)**
olumlu	11691	4529
olumsuz	23385	10350

Tablo 6.9'da görüldüğü üzere toplamda 16220 adet olumlu tweet, 33735 adet ise olumsuz tweet vardır. Her iki ortam(tek node'luk ve 4 node'luk) için sınıflandırma işlemi 5 defa yapılmış ve sonuçların bir birine çok yakın olduğu görülmüştür, bu nedenle bu bölümde hesaplamaların daha kolay yapılması açısından tablolara uygulamanın bir çalıştırma işleminin sonucundaki sınıflandırma değerleri yazılmıştır. Bu değerler tek node'luk Hadoop kümesi üzerindeki çalıştırma işleminin sonuçlarıdır.

Mahout üzerinde Tablo 6.9'da bahsedilen veri kümesi çalıştırılmış ve sınıflandırmanın karmaşıklık matrisi Tablo 6.10'da görüldüğü gibi olmuştur. Karmaşıklık matrisi modelin sınıflandırma değerlendirme sonucundan oluşmaktadır. Tablo 6.10 incelendiğinde olumlu bir tweet olup olumlu olarak sınıflandırılan(True Pozitif) 2776 tane özellik vektörü vardır. Olumlu olup olumsuz olarak sınıflandırılan(False Negatif) 436 tane özellik vektörü vardır. Benzer şekilde olumsuz olup yanlış bir şekilde olumlu olarak sınıflandırılan(False Pozitif) 1755 tane, olumsuz olup olumsuz olarak sınıflandırılan(True Negatif) 4983 tane kayıt vardır.

Tablo 6.10: Uygulama-2 karmaşıklık matrisi

	Olumlu	**Olumsuz**
Olumlu	2776	436
Olumsuz	1755	4983

Sınıflandırma modelinin doğruluk oranı aşağıdaki Denklem 6.1'deki formülle hesaplanmaktadır.

$$Doğruluk = \frac{True\ Pozitif + True\ Negatif}{\left(True\ Pozitif + False\ Pozitif + True\ Negatif + False\ Negatif \right)} \quad (6.1)$$

Bu uygulama sonuçları için doğruluk değeri manuel olarak hesaplamak istediğimizde;

$$Doğruluk = \frac{2776+4983}{(2776+1744+436+4983)} = \frac{7759}{9939} \cong 0,7797$$ şeklindedir.

Bu uygulama içi mahout tarafından hesaplanan doğruluk değerin de Şekil 6.8'de görüldüğü üzere % 77.97 olarak hesaplamıştır. Yapılan hesaplamalarda virgülden sonraki dört basamak kullanılmıştır. F1 değerinin Hassasiyet(Precision) ve Getirim(Recall) ile ilgili olduğu söylenmişti. Hassasiyet ve getirim değerleri aşağıdaki gibi hesaplanmaktadır.

$$Hassasiyet = \frac{True\ Pozitif}{(True\ Pozitif + False\ Pozitif)} \tag{6.2}$$

$$Getirim = \frac{True\ Pozitif}{(True\ Pozitif + False\ Negatif)} \tag{6.3}$$

$$F1 = \frac{2xHassasiyetxGetirim}{Hassasiyet + Getirim} \tag{6.4}$$

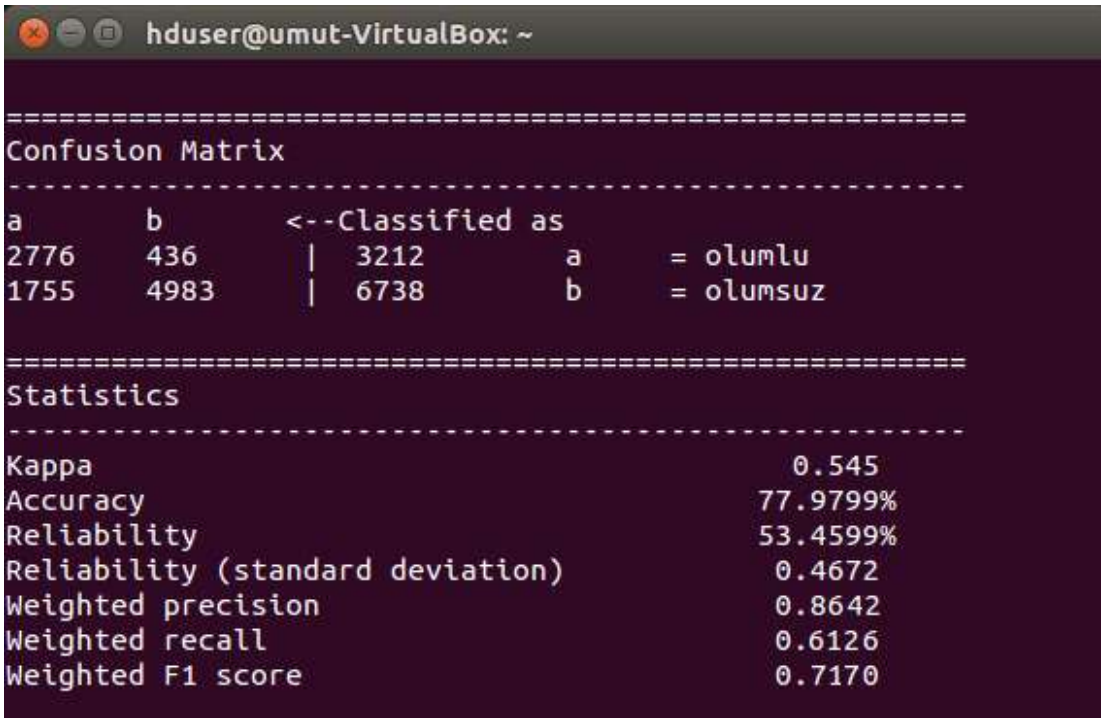
Yukarıdaki Denklem 6.2, 6.3 ve 6.4 doğrultusunda F1 değerini manuel olarak hesaplamak gerekirse;

$$Hassasiyet = \frac{2776}{(2776+1755)} \cong 0,6126$$ olarak bulunur.

$$Getirim = \frac{2776}{(2776+436)} \cong 0,8642$$ olarak bulunur.

$$F1 = \frac{2x0,8642x0,6126}{0,8642+0,6126} \cong 0,7170$$ olarak bulunur.

Veri kümesinin mahout üzerinde sınıflandırma işlemi sonrasındaki diğer istatistikleri Şekil 6.8'deki ekran görüntüsünde olduğu gibidir.



Şekil 6.8: Uygulama-2 sınıflandırma sonuçları

Aynı veri kümesinin 4 node'luk Hadoop kümesi üzerindeki sınıflandırma işlemi sonucu olarak ortalama doğruluk değeri 78.120 olarak bulunmuştur. Ortalama F1 değeri ise 0.7812 olarak hesaplanmıştır.

Aynı veri kümesinin 4 node'luk Hadoop kümesi üzerinde sınıflandırılması için verilerin HDFS dosya sistemine yüklenmesi zaman almıştır. Mahout'ta Naive Bayes algoritması yardımıyla yapılan sınıflandırma işleminde tek node'luk sistem üzerindeki işlemlerin süreleri ile 4 node'luk Hadoop kümesi üzerindeki işlemlerin süreleri Tablo 6.11'de olduğu gibidir.

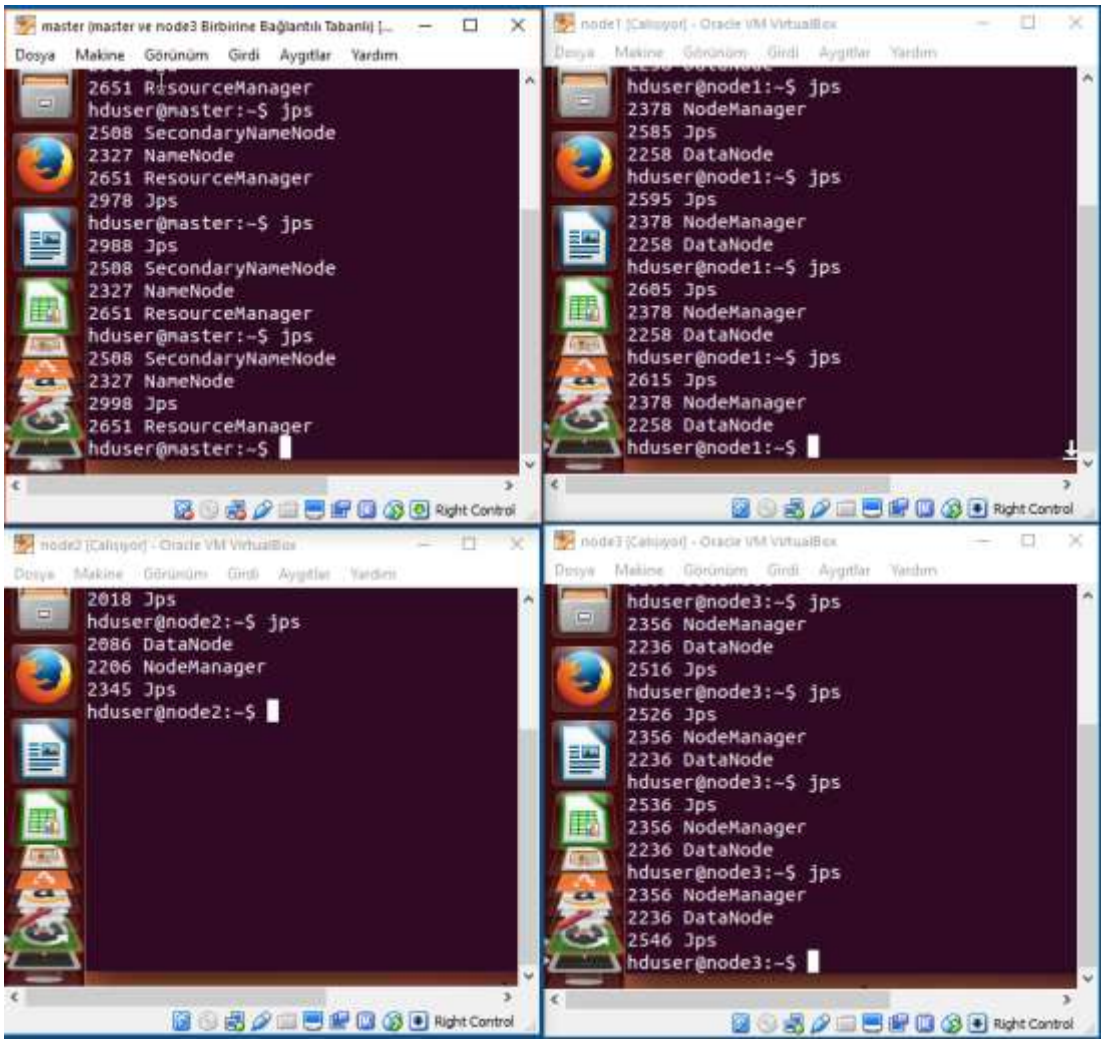
Tablo 6.11: Uygulama-2 çalışma süreleri

İşlem	Tek Node'luk Hadoop(dakika)	4 Node'luk Hadoop(dakika)
SequenceFile Dosyası Oluşturma	2.089	1.889
Vektör Oluşturma	0.851	6.897
Algoritma Eğitimi	0.278	1.973
Algoritma Testi	0.167	1.618

Hadoop büyük boyutlardaki veri kümelerini işlemek için geliştirilmiş bir kütüphanedir. Tablo 6.11'de görüldüğü gibi küçük boyuttaki dosyalar için küme

üzerinde çalışmak dezavantaj gibi görülmektedir. Gazetelere ait tweet'leri saklayan metin dosyalarının boyutları 1-2 kilobayt düzeyindedir. Bu nedenle tek node'luk sistemin hesaplama süreleri 4 node'luk sisteminkine göre daha kısadır. Kullanılan Hadoop sistemlerinde veriler 64 megabaytlık alt bloklara ayıracak şekilde ayarlanmıştı. Kullanılan metin dosyalarının boyutları 64 megabaytı geçmediğinden HDFS dosya sisteminde her bir metin için bir blok oluşturulmuştur. Her blok için 2 tane de kopya diğer node'larda saklandığından, küçük boyutlu bir dosya için çok fazla işlem yapılmıştır. Dosya sayısının da çok olması nedeniyle işlemler için harcanan süre artmıştır. Bu nedenle 4 node'luk sistemin hesaplama süreleri daha yüksektir. Bunların yanında sanal makinalar üzerinde çalışıldığından kümedeki bilgisayarlara paylaştırılan kaynaklar kısıtlı olması ayrıca hesaplama sürelerini etkilemiştir.

4 node'tan oluşan Hadoop sisteminin çalışma sırasındaki bir görüntüsü Şekil 6.9'da görüldüğü gibidir. Node-1, node-2 ve node-3 bilgisayarları datanode olarak yani slave olarak çalışmaktadırlar. Master olarak çalışan bilgisayar ise kümenin yöneticisi olarak çalışmaktadır. Jps komutu ile java sanal makinalar üzerinde çalışan Hadoop bileşenlerin durumları görüntülenebilir. Şekil 6.9'da görüldüğü üzere master node üzerinde, ResourceManager, NameNode ve SeconderyNameNode bileşenleri çalışmaktadır. Diğer datanode'ları için NodeManager ve DataNode bileşenleri çalışmaktadır.



Şekil 6.9: Hadoop kümesi çalışma zamanı

Bu uygulamanın ikinci bölümünde tek node'luk ve dört node'luk Hadoop sistemlerinin çalışma performansları incelenecektir. Çok sayıdaki küçük metin dosyaları birleştirilerek daha büyük boyutlu ve az sayıda metin dosyası oluşturulacaktır. Her bir tweet'i saklayan küçük boyuttaki metin dosyaları, olumlu ve olumsuz sınıfı için rastgele 100 tanesi bir metin dosyası olacak şekilde birleştirilmiş ve küme üzerinde çalıştırılmıştır. Yani 100 adet olumlu tweet bir tane metin dosyası olacak şekilde birleştirilmiştir. Bu metin dosyası olumlu sınıfındaki bir dokümandır. Birleştirme işleminde her metin dosyasındaki metinler aralarında bir boşluk olacak şekilde bir birine eklenmiştir. Bu şekilde sınıflandırma işleminin alt adımlarının hesaplanma sürelerindeki değişim gözlemlenmek istenmiştir. Birleştirme işleminden sonra her sınıftaki doküman sayıları Tablo 6.12'deki gibidir. Birleştirme işleminden sonra her bir dokümanın boyutu 6 ile 15 kilobayt arasında değişmiştir.

Tablo 6.12: Uygulama-2 birleştirme sonrası eğitim ve test metin sayıları

	Eğitim Verisi(%66)	**Test Verisi(%33)**
olumlu	116	45
olumsuz	233	103

Tablo 6.12'deki veri kümesi tek node'luk ve dört node'luk Hadoop kümeleri üzerinde çalıştırılmıştır. Birleştirme işleminden sonra toplamda 161 tane olumlu tweet'leri saklayan metin dosyası ve olumsuz tweet'leri içeren 336 tane metin dosyası oluşturulmuştur Birleştirme işleminden önce toplamda 49955 adet metin dosyası vardı, birleştirme işleminden sonra bu rakam 497 olmuştur. Mahout üzerinde eğitim verisi 5 defa çalıştırılmış ve her işlem için harcanan ortalama süreler Tablo 6.13'teki gibidir. Genel olarak bütün işlemler için çalışma sürelerinin azaldığı görülmektedir.

Tablo 6.13: Uygulama-2 birleştirme sonrası işlem süreleri

İşlem	**Tek Node'luk Hadoop(dakika)**	**4 Node'luk Hadoop(dakika)**
SequenceFile Dosyası Oluşturma	0.176	0.799
Vektör Oluşturma	0.759	6.102
Algoritma Eğitimi	0.456	1.776
Algoritma Testi	0.142	0.466

Dört node'luk küme için en büyük değişiklik algoritmanın test edilmesi süresi olmuştur. Bu süre yaklaşık 1.6 dakikadan 0.4 dakikaya düşmüştür. Tek node'luk Hadoop sistemi içinde en büyük zaman farkı sequencefile dosyası oluşturma adımında görülmüştür. Bu adımdaki işlem süresi yaklaşık 2.0 dakikadan 0.1 dakikaya düşmüştür. Birleştirme işleminden sonra oluşturulan metin dosyalarının boyutları yine 64 megabayttan daha düşük değerlerdedir. Dosya boyutunun 64 megabayttan daha yüksek ve dosya sayısının yüz binlerce olması durumunda, tek node'luk Hadoop ile küme üzerinde çalışan Hadoop sistemlerinin Tablo 6.13'teki işlemler için harcanan çalışma süreleri arasındaki fark daha da artacaktır.

Bu uygulamayla olumlu ve olumsuz olarak hazırlanan veri kümesi, bir kısmı eğitim bir kısmı da test verisi olarak iki farklı Hadoop ortamı üzerinde Mahout ile sınıflandırılmıştır. Sınıflandırma işlemiyle gazeteler tarafından atılmış olan tweet'ler

üzerinde duygu analizi yapılmıştır. Sınıflandırma başarım ölçütleri incelenmiş ve her iki ortamdaki başarımlar değerlendirilmiştir.

Yapılan uygulamanın ikinci kısmında ise tek node'luk Hadoop sistemi ile dört node'luk Hadoop sisteminin farklı boyuttaki ve farklı sayıdaki metin dosyalarını işlemek için harcadığı süreler incelenmiştir. Metin dosyası formatında saklanan tweet'leri sınıflandırmak amacıyla her bir Mahout komutu için harcanan süreler kaydedilmiş ve karşılaştırılmıştır.

7. SONUÇ VE ÖNERİLER

Bu tez çalışmasında büyük veri araçlarından Hadoop kullanılarak metin dosyaları üzerinde veri madenciliği ve doğal dil işleme yöntemleriyle duygu analizi yapılmıştır. Hadoop için iki ayrı gerçekleştirim ortamı oluşturulmuştur. Yapılan uygulamalarda Türkiye'deki 15 günlük gazetenin Twitter üzerinden belli tarihler arasında paylaşmış oldukları haber başlıkları sınıflandırılmıştır. Sınıflandırma işleminde Hadoop üzerinde çalışan Mahout aracı kullanılmıştır. Sınıflandırma algoritmalarından metin madenciliğinde en çok kullanılan ve en çok başarım elde edilen Naive Bayes algoritması kullanılmıştır.

Java programlama diliyle yazılan bir uygulama ile Twitter API'sinden gazeteler ait tweet'ler alınmıştır. API'nin kısıtlamalarından dolayı uygulama belli periyotlarla çalıştırılmıştır. Alınan tweet'ler metin dosyası olarak kaydedilmiştir. Bu tweet'ler olumlu ve olumsuz sınıflara ayrılmıştır. Bu işlemi gerçekleştirmek için yine java programlama diliyle eğitim verisi oluştur isimli bir başka uygulama yazılmıştır. Bu uygulama yardımıyla eğitim verisi oluşturulmuştur. Uygulama bir sınıfa dair işaretlenen bir tweet için metin madenciliği ve Türkçe doğal dil işleme yöntemlerini uygulamıştır. Türkçe doğal dil işlemleri için zemberek kütüphanesi kullanılmıştır.

Sınıflandırma işlemlerinde birinci uygulamada haber kategorisine göre haberin hangi gazeteye ait olduğuna dair sınıflandırma işlemi yapılmıştır. Eğitim ve test veri kümeleri için 3 gazete seçilmiştir. Oluşturulan sınıflandırma modeli %78 başarım sağlamıştır. Yani hangi gazeteye ait olduğu bilinmeyen bir haber başlığını % 78 doğruluk oranıyla doğru sınıfa atamaktadır.

Duygu analizi için yapılan sınıflandırma işleminde yani ikinci uygulamada ise haber başlıklarının olumlu veya olumsuz olarak sınıflandırmak istenmiştir. Oluşturulan eğitim ve test verileri iki farklı Hadoop ortamında çalıştırılmıştır. Oluşturulan sınıflandırma modeli yaklaşık %79 başarı ile yeni gelen bir tweet'i doğru sınıfa atamaktadır. Bu uygulamanın ikinci kısmında dosya boyutunun ve dosya sayısının yapılan sınıflandırma işlemi adımları üzerindeki çalışma zamanı incelenmiştir. Küçük boyutlu metin dosyaları(tweet'ler) birleştirilerek iki farklı Hadoop ortamında çalıştırılmış, her komut için harcanan zamanlar tablolarla ifade edilmiştir. Genel olarak küçük boyutlu ve çok sayıda dosya için harcanan zaman tek

node'luk Hadoop kümesi üzerinde daha az zaman alırken, dört node'luk Hadoop kümesi üzerinde daha çok zaman almıştır. Büyük boyutlu dosyalar içinse dört node'luk Hadoop kümesi genel olarak daha iyi performans sağlamıştır.

Bu tez çalışmasıyla büyük veri araçlarından Hadoop incelenmiş, büyük veriden anlamlı bilgiler çıkarmak için metin madenciliği yöntemi kullanılmıştır. Hadoop için yazılmış olan makine öğrenmesi yöntemlerini barındıran Mahout aracı incelenmiş ve kullanılmıştır. Bir veri kümesinden anlamlı bilgiler çıkarma uygulamalarından biri olan ve günümüzde popüler olan duygu analizi yöntemi kullanılmıştır. Duygu analizi için metin dilinin analizi için doğal dil işleme yöntemlerinden faydalanılmıştır. Genel olarak bu tez çalışması bir biri ile alakalı birden fazla disiplinin bütünleşik olarak kullanıldığı bir çalışma olmuştur. Çalışmanın Hadoop ortamında duygu analizi yapması çalışmanın özgün niteliklerindendir.

Yapılan bu çalışmayla büyük veri problemi duygu analizi çalışma bakış açısıyla değerlendirilmiştir. Büyük veri üzerinde duygu analizi gerçekleştirilerek, bir toplumun veya bir ülkenin herhangi bir konudaki fikirleri/duyguları tespit edilebilir. Bu şekilde toplumsal problemleri daha iyi değerlendirme ve daha hızlı çözüm üretme gibi fırsatlar elde edilebilir.

8. KAYNAKLAR

Akba, F., Uçan, A., Sezer, E., & Sever, H., "Assessment of feature selection metrics for sentiment analyses: Turkish movie reviews", *8th European Conference on Data Mining,* 180-184,(2007).

Akbaş, E., "Aspect Based Opinion Mining on Turkish", Yüksek Lisans Tezi, Bilkent Üniversitesi Fen Bilimleri Enstitüsü, Ankara, (2012).

Aktaş, Ö., "Türkçe için Verimli bir Cümle Sonu Belirleme Yöntemi", *Akademik Bilişim 2006,* Denizli, (2006).

Allen, R., What happens online in 60 seconds? SmartInsights[online], (18 Nisan 2016), http://www.smartinsights.com/internet-marketing-statistics/happens-online-60-seconds/

Altan, Z., & Orhan, Z., "Anlam Belirsizliği İçeren Türkçe Sözcüklerin Hesaplamalı Dilbilim Uygulamalarıyla Belirginleştirilmesi", *XX. Ulusal Dilbilim Kurultayı,* İstanbul, (2005).

Apache Spark[online], (26 Nisan 2016), https://en.wikipedia.org/wiki/Apache_Spark

Aşlıyan, R., & Günel, K., "Metin İçerikli Türkçe Dokümanların Sınıflandırılması", *XII. Akademik Bilişim*, Muğla, 529-535, (2010).

Aşlıyan, R., Günel, K., & Filiz, A., "Türkçe Otomatik Heceleme Sistemi ve Hece İstatistikler", *Akademik Bilişim 2006,* Denizli, (2006)

Aydın, G., & Hallaç, İ., "Document Classification Using Distributed Machine Learning", *International Conferance on Advances in Information Processing and Communication Technology* ,166-169, (2015).

Beyhan, H., "Sosyal Medya Üzerinde Metin Madenciliği ve Duygu Analizi ile Pazar Değerlendirme", Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, (2014).

Big Data Statistics (12 Şubat 2016). A Comprehensive List of Big Data Statistics[online], ,http://wikibon.org/blog/big-data-statistics/

Boynukalın, Z., "Emotion Analysis of Turkish Texts by Using Machine Learning", " Master's Thesis", Middle East Technical University, Ankara, (2012).

Chen, M., Mao, S., & Liu, Y., "Big data: a survey", *Mobile Networks and Applications*, 19.2, 171-2019, (2014).

Coenen, F. Data, "Mining: Past, Present and Future", *The Knowledge Engineering Review*, 25-29, (2004).

Çakmak, O., Kazemzadeh, A., Can, D., Yıldırım, S., & Narayanan, S., "Root-word analysis of Turkish emotional language", *Corpora for Research on Emotion Sentiment & Social Signals*, (2012).

Çetin, M., & Amasyalı, M. F., "Supervised and Traditional Term Weighting Methods", *In Signal Processing and Communications Applications Conference*, (2013).

Data Mining[online], (5 Mayıs 2016), https://en.wikipedia.org/wiki/Data_mining#Process

Dean, J., & Ghemawat, S.,"MapReduce: simplified data processing on large clusters", *Communications of the ACM*, 51.1, 107-113, (2008).

Demir, İ., "Hadoop Tabanlı Büyük Ölçekli Görüntü İşleme Altyapısı", Yüksek Lisans Tezi, Kocaeli Üniversitesi Fen Bilimleri Enstitüsü, Kocaeli, (2012).

Dikici, E., & Saraçlar, M., "Sliding text recognition in broadcast news.", *IEEE 16th Signal Processing, Communication and Applications Conference*, (2008).

Dolgun, M., Özdemir, T., & Oğuz, D., "Veri madenciliğinde yapısal olmayan verinin analizi: Metin ve web madenciliği", *İstatistikçiler Dergisi: İstatistik ve Aktüerya 2.2*, 48-58, (2009).

Döven, S., “Metin Madenciliği İle Dokümanlar Arasındaki Benzerliklerin Bulunması”, Yüksek Lisans Tezi, Bahçeşehir Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, (2013).

Ergün, K., ”Metin Madenciliği Yöntemleri ile Ürün Yorumlarının Otomatik Değerlendirilmesi”, Doktora Tezi, Sakarya Üniversitesi Fen Bilimleri Enstitüsü, Sakarya, (2012).

Eroğlu, U.,” Sentiment analysis in Turkish”, Yüksek Lisans Tezi, Orta Doğu Teknik Üniversitesi Fen Bilimleri Enstitüsü, Ankara, (2009).

Erten, F., “Metin Madenciliği Tabanlı Bir Web Sitesi Sınıflandırma Aracı Tasarımı” ,Yüksek Lisans Tezi, Maltepe Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, (2015).

Esteves, R., Pais, R., & Rong, C., “K-means clustering in the cloud-a Mahout test”, *Advanced Information Networking and Applications (WAINA),* 514-519, (2011).

Flink[online], (25 Nisan 2016), https://en.wikipedia.org/wiki/Apache_Flink

Flink-2[online], (20 Nisan 2016). http://www.infoworld.com/article/2919602/hadoop/flink-hadoops-new-contenderfor-mapreduce-spark.html

Flink-3[online], (20 Mart 2016), https://flink.apache.org/

Flume [online], (14 Mart 2016), http://flume.apache.org/

Gantz, J., & Reinsel, D., “Extracting value from chaos”, *IDC view 1142,* (2011).

Ghemawat, S., Gobioff, H., & Leung, S.-T., “The Google file System”, *ACM SIGOPS operating systems review,* Vol. 37, 29-43, (2003).

Güngör, O., & Güngör, T., “Türkçe bir sözlükteki tanımlardan kavramlar arasındaki üst-kavram İlişkilerinin Çıkarılması”, *Akademik Bilişim Konferansı,* 1-13, (2007).

Hadoop[online], (9 Mart 2016), Wikipedia: https://en.wikipedia.org/wiki/Apache_Hadoop

Hadoop Ecosystem [online], (2 Nisan 2016), https://hadoopecosystemtable.github.io/

Hadoop Kurulum [online], (12 Haziran 2016), http://www.michaelnoll.com/tutorials/running-hadoop-on-ubuntu-linux-single-node-cluster/

Hadoop Mode [online], (12 Haziran 2016), https://hadoop.apache.org/docs/r2.7.2/hadoop-project-dist/hadoopcommon /SingleCluster.html

Hammond, K., & Varde, A., "Cloud based predictive analytics: text classification, recommender systems and decision support", *2013 IEEE 13th International Conference on Data Mining Workshops*, 607-612, (2013).

Hbase [online], (12 Nisan 2016), https://en.wikipedia.org/wiki/Apache_HBase

HDFS Design [online], (3 Mayıs 2016). https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

History Data Mining [online], (5 Mayıs 2016), http://www.forbes.com/sites/gilpress/2013/05/28/a-very-short-history-of-datascience/#271178ac69fd

Hive[online], (12 Mayıs 2016), http://hive.apache.org/

İlhan, S., Duru, N., Karagöz, Ş., & Sağır, M., "Metin Madenciliği ile Soru Cevaplama Sistemi", EMO, 356-359, (2008).

JobTracker [online], (10 Mart 2016), https://wiki.apache.org/hadoop/JobTracker

Karaca, M., "Metin Madenciliği Yöntemi ile Haber Sitelerindeki Köşe Yazarlarının Sınıflandırılması", Yüksek Lisans Tezi, Karabük Üniversitesi Fen Bilimleri Enstitüsü, Karabük, (2012).

Karadağ, A., & Takçı, H., "Metin madenciliği ile benzer haber tespiti", *Akademik Bilişim*, Muğla, (2010).

Kaya, M., Fidan, G., & Toroslu, İ.,"Sentiment Analysis of Turkish Political News", *Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology*, Volume 01. IEEE Computer Society, 174-180, (2012).

Kaya, M., Fidan, G., & Toroslu, İ. "Transfer Learning Using Twitter Data for Improving Sentiment Classification of Turkish Political News", *In Information Sciences and Systems*, 139-148, (2013).

Kayım, F., "K-Means i le DBSCAN Algoritmasi'nin Paralelleştirmesi ve Hadoop Üzerinde Büyük Veri Analizinde Kullanılması, Performans ve Yeterlilik Karşılaştırması", Yüksek Lisans Tezi, Beykent Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, (2015).

KDD and Data Mining [online], (5 Mayıs 2016), http://www.rithme.eu/?m=resources&p=kdprocess&lang=en

Kuzucu, K., "Müşteri Memnuniyeti Belirlemek için Metin Madenciliği Tabanlı Bir Yazılım Aracı", Yüksek Lisans Tezi, Maltepe Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, (2015).

Mahout Kurulum[online], (13 Haziran 2016). http://androidyou.blogspot.com.tr/2011/11/mahout-and-hadoop-are-all-java.html

Mahout Parametreler[online], (14 Haziran 2016), https://mahout.apache.org/users/classification/twenty-newsgroups.html

Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., et al. "Big data: the next frontier for innovation, competition, and productivity." McKinsey & Company, (2011).

Map-reduce[online], (3 Mayıs 2016), http://sci2s.ugr.es/BigData

Mayda, İ., & Aytekin, Ç., "Sosyal Medyada Rekabet Analizi için Karşılaştırma Görevine Yönelik Fikir Madenciliği Modeli", *Journal Academic Marketing Mysticism Online*, vol. 7, no. 27, 414-425, (2013).

Meral, M., & Diri, B. “Twitter Üzerinde Duygu Analizi”, *2014 IEEE 22nd Signal Processing and Communications Applications Conference (SIU 2014)*, (2014).

Mitchell, R., What happens in one minute on the internet [online], (25 Nisan 2016), http://www.conservativedailynews.com/2016/04/what-happens-in-oneminute-on-the-internet-chart/

Nizam, H., & Akın, S., “"Sosyal Medyada Makine Öğrenmesi ile Duygu Analizinde Dengeli ve Dengesiz Veri Setlerinin Performanslarının Karşılaştırılması”, *XIX. Türkiye'de İnternet Konferansı*, (2014).

O’Malley, O. “Terabyte sort on apache hadoop”, (2008).

Oğuzlar, A., *Temel Metin Madenciliği*, 15366, Bursa: Dora Yayınları, (2011).

Özkaya, S., & Diri, B., “Türkçe Metinlerde Şartlı Rastgele Alanlarla Varlık İsimi Tanımı”, *IEEE 19th Signal Processing and Communications Applications Conference,* (2011).

Pig[online], (12 Nisan 2016), https://cwiki.apache.org/confluence/display/PIG/Index;jsessionid=BF0C00B3CC8389C308BC01D5E5523417

Sahu, L., & Mohan, B., “An improved K-means algorithm using modified cosine distance measure for document clustering using Mahout with Hadoop”, *Industrial and Information Systems (ICIIS),* 1-5, (2014)

Shafer, J., Rixner, S., & Cox, A., “The hadoop distributed filesystem: Balancing portability and performance”, *Performance Analysis of Systems & Software (ISPASS),* 122-133, (2010).

Shvachko, K., Kuang, H., Radia, S., & Chansler, R., “The Hadoop Distributed File System “, *In 2010 IEEE 26th symposium on mass storage systems and technologies (MSST) IEEE*, 1-10, (2010).

Silahtaroğlu, G., *Veri Madenciliği Kavram ve Algoritmaları*, 11968, İstanbul: Papatya Yayıncılık Eğitim, 2013.

Spark[online], (18 Nisan 2016), http://www.mammatustech.com/introduction-toapache-spark

Şimşek, M. U., & Özdemir, S., "Analysis of the relation between Turkish twitter messages and stock market index", *Application of Information and Communication Technologies (AICT) 2012 6th International Conference on. IEEE*, (2012).

Tajo[online], (12 Nisan 2016). http://tajo.apache.org/

Thelwall, M., Buckley, K., Paltoglou, G., Cai, D., & Kappas, "Sentiment strength detection in short informal text." , *Journal of the American Society for Information Science and Technology*, 2544-2558, (2010).

Twitter4j[online], (14 Haziran 2016), http://twitter4j.org/en/index.html

Uçan, A., "Otomatik Duygu Sözlüğü Çevirimi ve Duygu Analizinde Kullanımı", Yüksek Lisans Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, Ankara, (2014).

Vektör uzay modeli[online], (10 Haziran 2016),
http://www.csharpnedir.com/articles/read/?id=731

Vural, A., Cambazoglu, B., Senkul, P., & Tokgoz, Z., "A framework for sentiment analysis in Turkish: Application to polarity detection of movie reviews in Turkish", Computer *and Information Sciences III*, 437-445, (2013).

Wikipedia[online],(14 Mayıs 2016),
https://en.wikipedia.org/wiki/Big_data#cite_note-Editorial-12

Yıldız, H., Gençtav, M., Usta, N., Diri, B., & Amasyalı, M., "Metin Sınıflandırmada Yeni Özellik Çıkarımı", *IEEE SIU 2007 15. Sinyal İşleme, İletişim ve Uygulamaları Kurultayı*, (2007).

Zaharia, M., Chowdhury, M., Franklin, M., Shenker, S., & Stoica, I., "Spark: cluster computing with working sets", *HotCloud*, 0-10, (2010).

Zemberek[online], (16 Haziran 2016), http://zembereknlp.blogspot.com.tr/ adresinden alınmıştır

9. ÖZGEÇMİŞ

Adı Soyadı	: Mehmet Umut SALUR
Doğum Yeri ve Tarihi	: Birecik- 25.12.1988
Lisans Üniversite	: Pamukkale Üniversitesi
Elektronik posta	: umutsalur@gmail.com
İletişim Adresi	: Harran Üniversitesi Bilgisayar Mühendisliği Bölümü Osmanbey Kampüsü/ Şanlıurfa

Yayın Listesi :

• Akaslan, D., **Salur, M. U.**, Tenekeci, M. E. & Gümüşçü, A. "Designing a Relational Database to Alleviate the Impacts of Foreign Words on the Turkish Language", International Journal of Social Sciences and Education Research, "1/2", 630-641 pp. Available Online: 17/12/2015.

Konferans listesi :

• Akaslan, D., Tenekeci, M. E. & Gümüşçü A. & **Salur, M. U.**, Designing a relational database to alleviate the impacts of foreign words on the Turkish language, International Conference on Social Sciences and Education Research (ICSER), 29-31 October 2015, Antalya, Turkey

• Akaslan, D., **Salur, M. U.**, Gümüşçü, A. & Tenekeci, M. E., Measuring the Prevalence of Foreign Originated Words on the National Newspapers using Search Engines "10th International Computer & Instructional Technologies Symposium (ICITS),, 16-18 May 2016, Rize, Turkey