T.C
SELÇUK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

WEB UYGULAMA VE SUNUCULARININ PERFORMANS ANALİZİ

İbrahim Berkan AYDİLEK

YÜKSEK LİSANS TEZİ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

Konya, 2006

T.C

SELÇUK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

WEB UYGULAMA VE SUNUCULARININ PERFORMANS ANALİZİ

İbrahim Berkan AYDİLEK

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Bu tez 25.07.2006 tarihinde aşağıdaki jüri tarafından oybirliği / oyçokluğu ile kabul edilmiştir

Prof.Dr.Ahmet ARSLAN	Doç.Dr.Şirzat KAHRAMANLI	Doç.Dr.Hakan IŞIK
(Danışman)	(Üye)	(Üye)

ÖZET

Yüksek Lisans Tezi

WEB UYGULAMA VE SUNUCULARININ

PERFORMANS ANALİZİ

İbrahim Berkan AYDİLEK

Selçuk Üniversitesi Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Ahmet ARSLAN

2006, 95 Sayfa

Jüri: Prof. Dr Ahmet ARSLAN

Doç. Dr. Şirzat KAHRAMANLI

Doç. Dr. Hakan IŞIK

Bu çalışma web sunucu ve uygulamalarını izleme, değerlendirme, sınıflama ve performans analizi tahmini amacıyla yapılmıştır. Bu amaçla kullanılan bazı istemci ve sunucularda işlemci, hafıza kullanım oranı, farklı web uygulama teknolojileri, web sunucusunun hizmet edebileceği maksimum istek sayıları kullanılmıştır. Uygulanan web uygulamasının bahsedilen özelliklerinin değişmesiyle randımanda ve performansta meydana gelen değişiklikler araştırılmış ve bunların değişmesiyle ortaya çıkacak uygulama performansı Bayes Teoremi ile tahmin edilmeye çalışılmıştır. Çalışmada 7 adet farklı özelliklerde web sayfası kullanılarak deneme ve test yapılmıştır.

Anahtar Kelimeler: web sunucu performansı, web uygulamaları performansı, Saf Bayes Teoremi

ABSTRACT

Master Thesis

PERFORMANCE TESTING OF

WEB APPLICATIONS AND WEB SERVERS

Selçuk University

Graduate School of Natural and Applied Sciences

Department of Computer Engineering

Supervisor: Prof. Dr. Ahmet ARSLAN

2006, 95 Page

Jury: Prof. Dr. Ahmet ARSLAN
Doç. Dr. Şirzat KAHRAMANLI
Doç. Dr. Hakan IŞIK

This study has been done for pursuing, scoring, classification of the web server and its practice and for prediction of performance analyzing. Therefore some different web applications used for get statistics from clients, web servers CPU, memory using percent, handled maximum requests per second. When the explained features changed in applicated web server were done the differences in the output and the performance were researched and the application performances after changes were made to predict with Naïve Bayes Teorem. In this study 7 different web pages, applications with different features were used for attempting and testing.

Key Words: web server performance, web application performance, Naive Bayes Classification

ÖNSÖZ

Hayatımıza birçok yenilik ve kolaylığı getiren alışılagelmişin dışında yeni ufuklar açan internetin gün geçtikçe kullanımı artmıştır. Bu artış neticesinde asıl internetin kendisini oluşturan web uygulamalarının ve web sunucularının önemi çok büyüktür, insanların internet üzerinden bilgiye ulaşım haklarının önüne çıkabilecek sorunları önceden çözebilmek ve sistemde oluşabilecek darboğazları gidermek genel anlamda eğitime ve insanlığa kendimce yapabileceğim bir katkı olarak görüyorum.

Bu tez çalışmasında bana büyük desteği ve katkısı olan danışmanım Prof. Dr. Ahmet ARSLAN' a ve Sakarya üniversitesi öğretim görevlileri Arş.Grv Tuğrul Taşçı, Öğr.Grv İbrahim Karadoğan'a teşekkür ederim.

İÇİNDEKİLER

SİMGELER

ADSL : Asymetric Digital Subscriber Line (Asimetrik Sayısal Abone Hattı)

ARPAnet :Advanced Research Project Agency Network (İleri Araştırma Projeleri Kurumu

Bilgisayar Ağı)

ASCII : American National Standard Code for information interchange (Veri değişimi

için, Amerikan ulusal kod standardı)

ASP Active Server Pages (AktifSunucu Sayfalan)

Bps : Bytes Per Second (Saniyede iletilen Byte Sayısı)

CAB : Cabinet File (Cabinet Dosyası)

CGI : Common Gateway Interface (Ortak Geçit Arayüzü)

DNS : Domain Name Server (Alan Adı Sunucusu)

DoD : Department Of Defence(Savunma Bakanlıgı):

DSN : Data Source Name (Veri Kaynagı ismi):

EBCDlC : Extended Binary Coded Decimal Interchange Code (Genişletilmiş ikilik kodlanmış onluk sayılar için değişim kodu)

FTP : File Transfer Protocol (Dosya Transfer Protokolü)

HTML : Hyper Text Markup Language (Yazıötesi işaretleme Dili)

http : Hypertext Transfer Protocol (Yazıötesi Transfer Protokolü)

IIS : Internet information Services (İnternet Bilgi Servisleri)

İP : Internet Protocol (İnternet Protokolu)

ISAPI : Internet Server Application Programming Interface (İnternet Sunucuları Uygulama Programlama Arayüzü)

ISO : International Organisation for Standardization (Uluslar arası Standartlar Enstitüsü)

JAR : Java Archive (Java arşiv dosyası)

JSP : Java Server Pages (Java Sunucu Sayfaları)

Km : Kilometre

L2 : Level 2 (İkinci Seviye)

LAN : Local Area Network (Yerel Alan Ağı)

Milnet : Military Network(Ordu Bilgisayar Ağı)

ms : Milisaniye

NT : New Technology(Yeni Teknoloji)

ODBC : Open DataBase Connectivity (Açik Veritabanı Bağlantısı)

Osi : Open System interconnection(Açık Sistem Bağlantısı)

PC : Personal Computer (Kişisel Bilgisayar)

PDU: Protocol Data Unit (Protokol Veri Birimi)

PHP: Hypertext PreProcessor (Yazıötesi Önişlemcisi)

Request/Sn : Web sunucusunun saniyede cevaplayabileceği istek sayisi

SGML : Standard Generalized Markup Language(Genelleştirilrniş Standart işaretleme Dili)

SQL : Structural Query Language (Yapısal Sorgulama Dili)

TCP/IP : Transfer Control Protocol/ Internet Protocol(İletim Kontrol Protokolü/İnternet Protokolü)

TTLB : Time To Last Byte (Son Byte Transfer Edilene Kadar Geçen Zaman)

UCLA : University of Califomia Los Angeles (Los Angeles' taki Kaliforniya Üniversitesi)

UCSB : University of California Santa-Barbara (Santa Barbara'daki Kaliforniya Üniversitesi)

URL : Unifom Resource Locator (Tektip Kaynak Yeri Göstericisi)

UTP : Unshielded Twisted Pair (Korumasiz Dolanmış Kablo Çifti)

WAN : Wide Area Network (Geniş Alan Ağı)

WAS : Web Application Stress tool (Web Uygulaması Baskı Aracı)

www : World Wide Web (Dünyayı Saran Örümcek Ağı)

XML : Extensible Markup Language (Genişletilebilir İşaretleme Dili)

1. GİRİŞ

1.2. Problemin Tanımı

Günümüzde internet uygulamalarının alt yapısını kullanarak hizmet veren sayfa sayısı her geçen gün artmaktadır. Bu sayfalar geliştirilirken yapması gereken işlemleri sorunsuz yapması hedeflenmektedir ama kullanıcıya bundan fazlası gerekmektedir bu web sayfasına erişecek kullanıcıların hız bakımından o sayfadan beklentileri farklı olabilmektedir. Web sunucusunun verimli çalışabilmesi için sistem kaynaklarının optimum seviyede kullanılması gerekmektedir. Aksi takdirde sunucu sisteminde darboğazlar oluşması kaçınılmaz olacaktır. Bu darboğazlar sunucu kaynaklarında olduğundan dolayı ya sunucunun hizmeti çok yavaşlamakta ya da sunucu aniden, zamansız kilitlenmeler sonucu durmaktadır bu ise çoğu zaman web uygulamasının yeterli ilgiyi ya da hedefleneni vermediği gerekçesiyle yayından kaldırılması ile son bulur.

Bunun yanında web uygulamaları optimize etmenin birçok faydası vardır bunlardan bazıları şöyle sıralanabilir, optimize yapılarak sunucunun değişmesi gerektiğine inanılan işlemci, anakart, hafıza gibi maliyetli parçalarının değişme süresi ertelenebilir ya da bu ihtiyaç ortadan kaldırılabilir ayrıca sunucu hizmeti maksimum yapılırken kullanıcıların bekleme süreleri minimuma indirilmektedir.

1.2. Amaç

Bu çalışmada değişik web uygulaması öğeleri içeren fakat temelde aynı işi yapmaya çalışan bir model oluşturulmuştur. Bu model kullanıcı tarafında çalışan statik sayfalarda Javascript ile sunucu tarafında çalışan uygulamalarda ise C# dilinde Asp.Net ile hazırlanmıştır.

Temelde web uygulaması geliştirme ve web sunucu optimizasyonu biraz sanat ve birazda fen bilimleri işi olarak değerlendirilmelidir. Bir ressam düşündüğü

resmi çizmeden önce duygularını karakalem çalışması olarak yansıtır tuvale.[13] Bir web uygulaması içinde aynı şeyler geçerli olabilir sunulacak ya da sunulmakta olan web uygulamasının performans analizi işin en başında yapmamız gerekir. Yaptığımız çalışma sonucunda kullanılacak sunucu kaynakları optimum kullanılacak mı ya da sunucu kaynaklarının kullanma ölçümleri neticesinde kullanıcılar tarafından uygulama performansı nasıl bulunacak en fazla kaç kullanıcıya aynı anda hizmet verebilir sorularının yanıtları ve olası değişiklerde sonuçları tahmini için fen bilimleri veri madenciliği konu ve tekniği Saf Bayes Teorisine göre çıkarımlar yapılabilir.

2. LİTERATÜR ARAŞTIRMASI

(Apte, V., 2003) FastCGI, CGI, Java Servlets, Java Server Pages gibi dinamik web programlama teknikleri işlemci kullanım ve hafıza kullanımına göre artan kullanıcı sayısı ile karşılaştırılmış sunucunun cevap/sn sayacı sonucları değerlendirme kullanılmıştır.

(Tuğ, E., 2006) Algoritma(GA) kullanılarak web log dosyalarından sıralı erişimelerin bulunması ile ilgilidir. Loglarda depolanan bilgiler internet kullanımının hızla artması sebebiyle kullanıcı davranışlarını keşfetmede oldukça önemli savunulmuş ve bu amaçla ardı ardına ziyaret edilen en iyi yol keşfedilmeye çalışılmıştır.

(Bai, G., 2006) Kablosuz ağlarda web sunucularının nasıl performans sergilediklerini incelenmiştir.Sorun çıkaran bileşenler özetlenmiştir, kullanıcı taraflı paket kayıpları, sunucu tarafındaki paket kayıpları ,ağ zafiyeti, sürekli bağlı kalan http bağlantıları kullanılarak mobil uygulamalarda kablosuz web sunucularına 350% artış sağlanmıştır.

(Lui , Z., 2005) Web uygulaması ve web sunucu için bir trafik modellemesi yapılmıştır,Bunun yanı sıra Http1.1 ile Http1.0 mukayese edilerek avantaj ve dezavantajları listelenmiştir. Yapılan web modellemesinde ziyaretçinin kullanım istatistikleri alınarak web sunucuya gönderilmektedir.Ziyaretçinin bir sonraki gelişinde ona uygun http protokolü kullanmaktadır.

(Kranakis, E., 2003)Bir web sayfasını ziyaret edilmesi ve devamında ziyaretçinin hangi sayfaya yöneldiği kayıt altına alınarak sayfalar arasında ağırlıklı-greedybfs algoritmasına göre linkler konulmuştur böylelikle web server transferinde ortalama %15 daha az veri transferi sağlanmıştır.

(Karadoğan, İ., 2003) Web tabanlı sistemlere dinamik ve sunucu tarafında çalışan öğelerin eklenmesi ve bu tür sistemlerin kullanımının artmasıyla birlikte beliren yavaşlama ve sistem kilitlenmesi gibi performans düşüklükleri bu sistemlere bağlanan kullanıcıları büyük oranda etkilemektedir. Bu çalışmada, Web tabanlı öğretim hizmetlerinde çok büyük oranlarda kullanılan sunucu taraflı uygulamalarla birlikte ortaya çıkan performans düşüklüklerinin nedenleri araştırılmış ve bunları önlemeye yönelik yapılacak çalışmalar belirlenmeye çalışılmıştır. Bu amaçla bir dizi test yapılmıştır ve test sonuçları değerlendirilmiştir.

(Squillante, Mark S., 1999) Çeşitli web sitelerini incelemek ve anlamak ve performans değerlendirmesi yapmak için kullanıcı kitlesinin ve yapısının iyi anlaşılması gerekir.

(Dilley, J., 1998) Güngeçtikce web sunuculara olan yük artmaktadır layered queuing models (LQMs) Adı verilen sistem ile kuyrukta bekleyen sayfaların sisteme etkilerini ve zamanlarını inceleyerek web sunucu da bulunan cache alanın artırılmasına dayalı bir çalışma yapılmıştır.

(Usal, M., 1998) Toplumların hedeflerine ulaşmada etkili bir araç olarak kullandıkları uzaktan eğitim modellerinin genel yapısı, iletişim ortamları, teknolojileri ve e-eğitimde etkili parametreler incelenerek, Türkiye örneği üzerinde durulmuştur.The Turkish Online Journal of Educational Technology – TOJET April 2005 ISSN: 1303-6521 volume 4 Issue 2 Article 6

(Iyengar, A., 1997) Web serverların performansı çok hacimli isteklere cevap veriyorsa kritik bir konudur bu çalışmada web sunucuların yüksek işlemci kullanımının araştırması yapılmıştır buna neden olan dinamik html sayfalarının az kullanılması tavsiye edilmiş ve çeşitli simülasyonlar yapılmıştır.

3. MATERYAL VE YÖNTEM

3.1 Giriş

Bu kısım tezin kapsamı gereği üç bölümden oluşmaktadır.

İlk bölümde web sunucuları ve web uygulamalarının kullanılabilmesi ve hizmet verebilmesi için gereken protokoller, standartlar ve temel kavramlardan bahsedilmeye çalışılacaktır. Bu kavram ve protokoller internet ve bilgisayar ağlarının altyapı temelleri anlamaktan geçmektedir. Bu nedenle internet, osi, ağ cihazları, isimlendirme, tcp/ip protokol grubu ve www hizmeti kısaca neler olduklarına değinilecektir.

İkinci bölümde web programlamadan bahsedilecektir. Web uygulamalarına ihtiyaçlar ve web programlarından beklentilerin zaman içerisinde değişimine paralel olarak web teknolojisindeki meydana gelen değişikliklerden bahsedilecektir. Statik durgun html sayfaları, Xml (genişleyebilir işaretleme dili), istemci taraflı script Javascript, Vbscript dilleri ve dinamik web programlama Asp, Asp.net ‘e değinilecektir.

Üçüncü bölümde veri madenciliği sınıflama ve tahmin yürütme konularından bahsedilecektir. Karar ağaçları, Saf Bayes teoremi, Bayes Ağları kavramlarından bahsedilecektir.

3.2 Materyal

Uygun test ortamını oluşturabilmek için web istemci bilgisayar, web sunucu bilgisayar ve gerekli yazılımlar gereklidir.

Web istemcisi olarak Pentium 4 3.2 Ghz işlemcili 1GB hafıza 100 Mbps bant genişliğine ile ağa bağlı donanım özellikleri bulunan bilgisayar kullanıldı. Bu istemci

Windows XP Professional ile çalışmaktadır. Test uygulama yazılımı olarak Microsoft firmasının geliştirdiği Web Application Stress Tool (WAS) programı kullanıldı. Bu program sayesinde bir web sunucusuna aynı anda istenen sayıda kullanıcı bağlantısı yapılarak gerçeğe yakın test sonuçları ve simülasyon yapılmasına imkan vermektedir.

Web sunucu bilgisayar olarak Pentium 4 3.2 Ghz işlemcili 1 GB hafıza ve 100 Mbps bant genişliği ile yerel ağa bağlı Windows 2003 server işletim sistemi ve IIS 6.0 (internet information server) web sunucusu kullanıldı.

Test için geliştirilen web sayfaları Visual Studio 7.0 C# ile ASP.NET kullanılarak hazırlanmıştır.

3.3 Yöntem

Bu bölümde internet, bilgisayar ağları, web programcılığı ve veri madenciliği sınıflama ve tahmin yöntemleri anlatılmaktadır.

3.3.1 İnternet ve Bilgisayar Ağları

3.3.1.1 İnternet tarihi

Internet ilk olarak ABD'de askeri amaçlı bir proje ile ortaya çıkmıştır. 1960'lı yıllarda soğuk savaş döneminin nükleer çatışma tehdidi yüzünden savunma amaçlı projelere büyük harcamalar yapılmaktaydı. ABD tarafından geliştirilen ve ARPANET (Advanced Research Projects Authority Net) adı verilen proje, ülke savunmasını birbirine bağlı bilgisayarlarla kurulacak iletişimle koordineli bir biçimde sağlamak amacıyla 1969 yılında geliştirilmiştir. Projeye göre herhangi bir bilgisayarın devre dışı kalması ağa bağlı diğer bilgisayarları etkilemeyecek ve iletişim devam edecekti. Ağı düzenleyen ya da denetleyen herhangi bir merkez bulunmadığından sürekli ve kesintisiz bir iletişim mümkün olabilecekti. Bugünkü Internet'in temelini oluşturan bu projede daha sonra aynı ağa başka yeni

bilgisayarların eklenmesiyle ağ üzerinden iletişim giderek arttı ve çok sayıda kullanıcının yararlandığı elektronik mektup, tartışma listeleri, forumlar, dosya transfer hizmetleri gibi yeni kullanım alanları ortaya çıktı. ARPANET'ten başka bilimsel amaçlı NSFNET (National Science Foundation) 1986 yılında, ticari amaçlı Compuserve gibi yeni ağlar da kullanıma açıldı. İlk olarak 1973 yılında birbirinden farklı ağların aralarında veri iletimi sağlayabilecekleri, ortak bir dil oluşturularak birleştirilmeleri kararlaştırıldı. Bu amaçla geliştirilen TCP/IP (Transmission Control Protocol / Internet Protocol) kullanılmaya başlandı. TCP/IP, Internet üzerinde yer alan farklı özellikte bilgisayarların ve ağların birbirleriyle sağlıklı bir şekilde iletişim kurabilmelerini sağlayan ortak bir dil olarak geliştirilmiştir. Internet'in gelişmesindeki son aşama ise WWW'in (World Wide Web) geliştirilmesidir. WWW Internet kullanımı ve kullanıcısının artmasında sağladığı kolaylıkla önemli bir işlevi yerine getirmiştir. [14]

3.3.1.2 İnternet'in tanımı

Internet birbiriyle tüm dünya üzerine yayılmış bilgisayar ağlarının birleşiminden oluşan devasa bilgisayar ağıdır. Internet'e bazıları ağların ağı da demektedir. Genellikle NET olarak adlandırılan Internet, bilgi otobanının şu anki açık bir şekillenmesi, vücut bulması olarak ortaya çıkmaktadır. Internet ile istenilen yere çabucak ulaşılabilir, dünyanın dört bir yanındaki bilgisayar sistemlerine mesajlar gönderilebilir. Bu mesajlar birkaç saniye içinde hedefe varır. Bir telefonla Internet'e bağlı bir bilgisayara bağlanılabilir, sisteme giriş yapılabilir, buradan da diğer Internet'e bağlı bilgisayarlara bağlanarak programlar çalıştırılarak kaynaklara ulaşılabilir. Internet'te ayrıca elektronik forumlarda diğer insanlarla çeşitli konularda iletişim kurulabilir, elektronik ilan panoları'na mesaj bırakılabilir, ağ haberleri gruplarına belirli konularda makaleler gönderilebilir. Internet'i insan vücuduna benzetebiliriz. Bunu insan vücudunun oluşturduğu biyolojik sisteme benzetebiliriz. Hücre ve organlar bir taraftan diğerleriyle iletişim kurarken bir yandan da kendi başına çalışır. Diğer organların sayısını, vücutla irtibat halinde olup olmadıklarını bilemezler. Herhangi bir hücre ölürse ve sistem dışında kalırsa diğer hücrelerin bundan haberi bile olmaz. Internet'e bağlı bilgisayar sistemleri de bu hücre ve

organlara benzer, bir taraftan kendi başlarına bağımsız olarak çalışmaya devam ederken bir taraftan da diğer sistemlerle iletişim kurarlar. Bunu yaparken Internet'e bağlı kaç tane bilgisayar var, diğer bilgisayarlar çalışıyor mu, Internet'e bağlantıları devam ediyor mu bilmeleri gerekmez. Başka bir bakış açısı da Internet'i bütün dünyayı saran otoyola benzetmektir. Internet'i sadece ana yol ve bu yola bağlı diğer yollar şeklinde düşünmek yanlıştır. Internet birbirine bağlı ve değişik genişlikte birçok yolun oluşturduğu sistemdir. Bu yolların üzerinde çeşitli büyüklüklerde park alanları mevcuttur. Bu yollardaki araçlar bilgi paketlerine, park yerleri ise bilgi depolarına benzetilebilir. "Internet nasıl bir arada duruyor" gibi bir soru akla gelebilir. Internet'in omurgasını oluşturan merkezi ya da çok büyük bilgisayar sistemlerine numara ataması yapan bir yönetim birimi vardır. Burası numaraları bloklar halinde sistemlere verir, numaraların parsellenmesi ve kendi içinde dağılımı bu sistemlerin yöneticileri tarafından yapılır. Internet'te merkezi yönetim birimi yoktur. Tüm her şeyin kaydını tutan merkezi makine olmadığından Internet'te toplam kaç bilgisayar olduğunu bilmenin imkanı yoktur. Her makineyi toplam kaç kişinin kullandığı bilinemeyeceği için de Internet'teki toplam kullanıcı sayısını tespit etmek daha da imkansızdır. Bilgisayardan başka makineye gönderilen bilgi yol boyunca muhtelif rotaları takip eder. Genelde bu olay kullanıcı tarafından fark edilmez, örneğin Internet üzerindeki bir hizmet makinesinden kendi bilgisayarınıza dosya aktarırken her iki taraf ta bu işlem esnasında verilerin hangi makinelerden geçtiğini bilmez. Bir elektronik mesaj çok uzunsa, birkaç parçaya ayrılır, Internet üzerinde değişik kanallardan hedefe ulaştırılır. Böylece tek bir kanalın sıkışması önlenir. Gönderen kişi ya da alıcı bunun farkına varmaz. Buna şeffaf teknoloji denir. Paketlerin aktarılma işlemini şöyle bir örnekle açıklamak mümkündür. İstanbul'daki bilgisayarın Erzurum'dan gönderilen ve Ankara'ya aktarılan bir paket aldığını varsayalım. İstanbul'daki bilgisayar bu durumda Erzurum'a paket göndereceği zaman bunu Ankara'ya gönderebileceğini, Ankara'nın Erzurum'un adresini bildiğini düşünür ve bu bilgiyi kendi hafızasındaki özel tabloya kaydeder. Daha sonra kendisine Erzurum'a gitmek üzere paket geldiğinde hafızasındaki tabloya bakarak bu paketi Ankara'ya göndermeye karar verir. Dikkat edilirse burada alıcının bilgisayarı ile arada birkaç sekme olmasına rağmen İstanbul'daki bilgisayar sadece komşu bilgisayarın (Ankara) adresini tutmaktadır. Aradaki tüm bilgisayarlar aynı şekilde

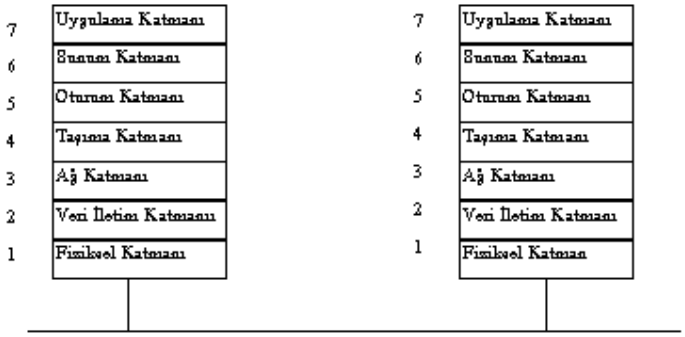
komşusunun adresini tuttuğu için paket alıcıya varır. Bilgisayarın tablosunda olmayan alıcı adresine sahip paket gelirse önceden saptanmış ve daha fazla yerin adresini tutma kapasitesine sahip bilgisayara gönderim yapılır. Bu aktarma sisteminin çalışabilmesi için işlemlerin çık hızlı yapılması gerekmektedir. Paket doğru yerde değilse derhal başka bilgisayara aktarılır. [14]

3.3.1.2 İletişim ağları yapısal modeli

3.3.1.2.1 OSI referans modeli

Bilgisayarlar arası iletişimin başladığı günden itibaren farklı bilgisayar sistemlerinin birbirleri arasındaki iletişim daima en büyük problemlerden birisi olmuş ve bu sorunun üstesinden gelebilmek için uzun yıllar boyunca çeşitli çalışmalar yapılmıştır. 1980'li yılların basında Uluslararası Standartlar Organizasyonu (International Standarts Organization-ISO) bilgisayar sistemlerinin birbirleri ile olan iletişiminde ortak bir yapıya ulaşmak yönünde çabaları sonuca bağlamak için bir çalışma başlatmıştır. Bu çalışmalar sonucunda 1984 yılında Açık Sistem Bağlantıları (Open Systems Interconnection-OSI) referans modeli ortaya çıkarılmıştır. Bu model sayesinde değişik bilgisayar firmalarının ürettikleri bilgisayarlar arasındaki iletişimi bir standarda oturtmak ve farklı standartlar arası uyumsuzluk sebebi ile ortaya çıkan iletişim sorununu ortadan kaldırmak hedeflenmiştir. OSI referans modelinde, iki bilgisayar sistemi arasında yapılacak olan iletişim problemini çözmek için yedi katmanlı bir ağ sistemi önerilmiştir. Bir başka deyişle bu temel problem yedi adet küçük probleme parçalanmış ve her bir problem için ayrı ayrı bir çözüm yaratılmaya çalışılmıştır. Bu yedi katmanın en altında yer alan iki katman yazılım ve donanım, üstteki beş katman ise genelde yazılım yolu ile çözülmüştür. OSI modeli, bir bilgisayarda çalışan uygulama programının, iletişim ortamı üzerinden başka bir bilgisayarda çalışan diğer bir uygulama programı ile olan iletişiminin tüm adımlarını tanımlar. En üst katmanda görüntü ya da yazı seklinde yola çıkan bilgi, alt katmanlara indikçe makine diline dönüşür ve sonuç olarak 1 ve 0'lardan ibaret elektrik sinyalleri halini alır. Şekil

3.1'de OSI referans modeli katmanları ve bir yerel ağ üzerindeki durumu gösterilmektedir: [15]



Şekil 3.1: OSI Referans modeli

OSI katmanlarının tanımlanan temel görevleri:

7- Uygulama Katmanı

Uygulama katmanı, kullanıcıların ağ üzerinde iletişim kurmaları için gerekli olan uygulamaları sağlar. Bu uygulama katmanı servislerinden bazıları şunlardır:

Elektronik posta iletimi: Elektronik posta alışverişi için kullanılan protokol kümelerini tanımlar. Bu sayede e-posta protokollerini kullanan tasarımcıların kendi e-posta iletim kurallarını belirlemelerine gerek kalmaz. Ortak e-posta ara yüzlerini kullanan uygulamalar, e-posta alışverişlerini yine bu ortak kuralları kullanarak yapabilirler.

Uzaktan dosya erişimi: Bölgesel uygulamalara, uzak düğümlerde bulunan dosyalara erişme imkanları tanınabilir.

Uzaktan uygulama çalıştırma: Bölgesel uygulamalara, uzak düğümlerde bulunan programlan başlatma ve kontrol etme imkanları sağlanabilir. Dizin hizmetleri: Ağdaki kullanıcılara, ağdaki kaynaklara sayısal ID' ler yerine mantıksal (logical) dizin adları oluşturarak kullanma imkanları tanınabilir.

Ağ yönetimi: Ağ yönetim protokolleri kullanılarak, uygulamalara, ağın yönetimsel araçlarına erişme ve bu araçları kullanma imkanları verilebilir. [6]

6-Sunum

Sunum katmanının görevi alt katmanlardan aldığı veriyi uygulama katmanına sunmaktır. Bazı durumlarda sunum katmanı veriyi bir formattan başka bir formata

direkt olarak dönüştürür. Örneğin IBM mainframe bilgisayarlar EBCDIC adı verilen bir kodlama şeması kullanırken diğer bilgisayarlar ASCII kodlama şeması kullanmaktadır. Eğer, EBCDIC 'i kullanan bir bilgisayar ile ASCII kodlarını kullanan bir bilgisayar arasında iletişim söz konusu olacaksa sunum katmanı, iletilecek olan verinin bu iki kod şeması arasında dönüşümünden de sorumlu olacaktır. Sayısal (numeric) verinin değişik bilgisayar mimarilerinde farklı şekillerde temsil edildiği düşünülürse, iletim sırasında bu verilerin karşı tarafta doğru şekilde alınmasını sağlayacak mekanizmaların da çalıştırılması gerekmektedir. Veri dönüşümünü sağlamak için kullanılan etkin bir yöntem, iletilecek olan verinin, standart bir formata dönüştürüldükten sonra iletilmesidir. Bu standart format kullanılmakta olan herhangi bir bilgisayarın formatı olmayabilir. Fakat iletim yapacak olan bilgisayarlar bu standart formattaki veriyi alacak ve kendi kullandıkları formata dönüştürecek şekilde yapılandırılabilir. Sunum katmanının diğer bir görevi ise- ihtiyaç duyulduğu zamanlarda verinin "şifrelenmesi "/"deşifre edilmesi" (encryption/decryption) "sıkıştırılması""açılması" (compression/decompression) olabilir. [6]

5- Oturum

Düğümler arasındaki diyalogları kontrol etmek oturum katmanının görevidir. Diyalog, iki düğümün veri alışverişi için anlaşmalarını sağlayan iletişim biçimidir. İletişim 3 farklı diyalog yöntemi ile gerçekleşebilir.

Simplex: Bağlı düğümlerden birisi sürekli veri ileticisi, diğeri ise veri alıcısıdır.
Half-duplex: İletişimin herhangi bir anında düğümlerden birisi verici diğeri alıcı durumunda bulunur. Daha sonra düğümler alıcı ve verici olarak değişirler.
Full-duplex: Düğümler eş zamanlı olarak hem veri gönderip hem de alabilirler. Full-duplex iletişim sırasında, alıcı ve verici düğümler arasındaki senkronizasyonu sağlamak amacıyla, devreye girmesi gerekir. Bazı akış kontrol mekanizmalarının Oturum katmanı tarafından idare edilen oturumların amacı iletişimi düzenli bir hale getirmektir.

Her oturumun 3 aşaması vardır.

Bağlantının kurulması: Düğümler, kullanılan protokol kümesinin bağlantı kurallarını ve iletişim parametrelerini kabul ederek bağlantı kurarlar.

Veri transferi: Düğümler veri alışverişinde bulunurlar.

Bağlantının koparılması: Düğümler veri iletişimini tamamladıktan sonra, oturumun kapatılması için anlaşırlar. [6]

4- Taşıma

Her ağ teknolojisi, ağ üzerinde iletilecek olan veri paketleri için bir maksimum sınır belirler. Örneğin Ethernet teknolojisinde veri alanlarının maksimum değeri olarak 1.500 Byte belirlenmiştir. Bu türden bir sınırlamaya gidilmesinin temel iki nedeni vardır:

i. Fazla sayıdaki cihazın ağı kullandığı durumlarda küçük çerçevelerin iletilmesi ağ verimliliğini arttırır. Eğer veri çerçevelerinin boyutlan sınırsız olsaydı, ağdaki tek bir cihaz bütün bant genişliğini kaplayacak şekilde veri iletimine başlayacak ve bu işlem bitene kadar diğer cihazlar ağda veri iletimi yapamayacaktı Küçük çerçeveler kullanılarak ağdaki her cihazın belirli zaman aralıklarıyla ağa ulaşması ve ağı kullanması sağlanır.

ii. Küçük çerçeveler kullanılarak, hata düzeltilmesi için mümkün olan en az boyutta veri tekrar iletilir. Örneğin 100 KByte' lık bir verinin bir bütün olarak iletilmesine izin verilen bir ağda bu 100 KByte' lýk verinin tek bir Byte'ında oluşan hatayı düzeltmek için 100 KByte' lık verinin tamamının yeniden iletilmesi gerekir. Eğer bu aynı veri 100 çerçeveye bölünerek iletilirse 1 Byte' lık hatayı düzeltmek için sadece 1 KByte' lık veriyi tekrar iletmek yeterli olacaktır.

İletim katmanının bir görevi iletilecek veriyi, ağda kullanılan teknolojinin belirlediği boyutlarda parçalara ayırmaktır. Alıcı taraftaki iletim katmanı ise parçalara ayrılmış bu veriyi birleştirerek orijinal veriyi elde eder. iletilen veri parçalara ayrılıp iletim ortamına bırakılırsa, alıcı tarafın bu veriyi iletildiği sıra ile alamama ihtimali kuvvetlidir. Bu nedenle gönderici tarafın iletim katmanı veriyi

parçalara bölerken her çerçeveye bir sıra numarası başlığı ekler, alıcı taraftaki iletim katmanı, çerçevelerin geliş sırası ne olursa olsun bu sıra numaraları ile orijinal veriyi tekrar oluşturur. [6]

3-Ağ Katmanı

Kullanılmakta olan bilgisayarların büyük çoğunluğu daha küçük bilgisayar ağlarının birleşmesinden oluşur. Küçük bilgisayar ağlarının oluşturduğu ağlara internetwork ya da internet adı verilir. Bu noktada internet ve İnternet kavramlarının farklı şeyler anlattığını açıklamak gerekmektedir. İnternet, milyonlarca bilgisayarın bağlı olduğu ve temel protokol kümesi TCP/IP olan bilgisayar ağını ifade ederken, internet ise kendine ait protokol kümesi bulunabilen ve küçük ağların birleşmesinden oluşan ağ yapısını ifade eder. Bu alt bölümlemeler sayesinde ağ, veri trafiği düşük olacak şekilde tasarlanabilir. Ayrıca ağa uzaktan ve daha yavaş olan ortamları kullanarak bağlanan ağlar izole edilebilir. Ağlar alt ağların birleşiminden oluşacak şekilde veri iletiminin boyutları değişir ve artık bölgesel ve küçük ağlar dahilinde iletilmekten çıkar. Bu durumda ağın genelinde verileri iletmek için bir yönlendirme mekanizmasının işlemesi gerekir.

Bir internetwork ' de verilerin iletilebilmesi için her alt ağın kendine ait bir ağ adresi ile temsil edilmesi gerekir. Ağ katmanı, iletilmek üzere üst katmanlardan aldığı veriye göndericinin ve alıcının ağ adreslerinin bulunduğu başlık bilgileri ekler. İletilecek veri ile bu başlık bilgilerinin birleşiminden oluşan veri birimine paket adı verilir. Verilerin iletilmesi için kullanılan ağ adresleri verinin doğru ağa ulaşmasını sağlar. Veri doğru ağa ulaştıktan sonra, veri bağı katmanı tarafından o ağda bulunan asıl alıcı düğüme ulaştırılır. Veri paketlerinin doğru ağlara iletilmesi, yönlendirme (routing) adını alırken veri yönlendirmesini yapan cihazlara yönlendirici (router) adı verilir.[6]

2-Veri bağlantı iletişim katmanı

Bir bilgisayar ağı üzerinde iletişim yapan cihazlara genel olarak düğüm (node) adı verilmektedir. Veri bağı katmanının temel görevi tek yerel bir ağda düğümden düğüme iletişim sağlamaktır. Bu servisi sağlamak için veri bağı katmanı şu iki fonksiyonu icra etmelidir: birinci fonksiyonu, verilerin doğru düğümlere iletilmesi için bir adresleme mekanizmasının sağlanmasıdır. İkinci fonksiyonu ise, fiziksel katmanın iletebilmesi için üst katmanlardan gelen veriyi bitlere çevirmesidir. Veri bağı katmanı iletmek için bir veri aldığında, bu veriyi çerçeve (frame) adı verilen birimlere dönüştürür. Çerçeve terimi yerıne paket (packet) de kullanılmaktadır. Yerel bir ağda çerçevelerin iletilmesi basit bir mantıkla yapılmaktadır. Veriyi gönderen düğüm çerçeveyi ağa bırakır. Ağda bulunan her düğüm bu çerçeveyi görür ve çerçevede bulunan hedef düğüm adresini kontrol eder. Eğer adres kendine ait ise adresin ait olduğu veri bağı katmanı veri çerçevesini alır[6]

1- Fiziksel katman

Bu katman ağın elektriksel ve mekanik karakteristiklerini belirler. Modülasyon teknikleri, çalışma voltajı, frekansı vs. bu katmanın temel özelliklerindendir. OSI referans modeli bir ağ uygulaması değildir. OSI sadece her katmanın görevini tüm detayları ile tanımlar. Bu modeli bir gemi ya da ev projesine benzetebiliriz. Nasıl ayni gemi planını alıp farklı firmalar gemi yapabilirse OSI modeli de böyledir. Nasıl ayni gemi planından iki farklı firma gemi ürettiğinde en azından kullanılan çiviler farklı yerlere çakılırsa, OSI modeli de gerçekleştiren firmadan firmaya farklılık gösterebilir.[15]

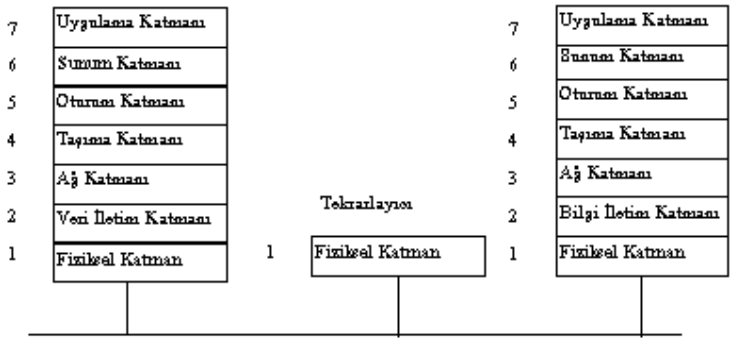
Fiziksel katman ağda kullanılan iletim ortamıyla direkt bağlantıya sahiptir ve iki görevi vardır: bitlerin gönderilmesi ve alınması. İkilik sistemde temsil edilen bir sayısal veri ya da bir bit, veri iletişiminin temel birimidir. İletim ortamının değişik durumlarını temsil etmek üzere yalnızca 1 ve 0 değerlerinden birini alabilir. Diğer iletişim katmanlarının görevleri ise bu bitleri, iletilen mesajı oluşturmak üzere bir araya getirmektir.[6]

3.3.1.3 Bağlantı aygıtları

Bilgisayar ağı erişiminde genel olarak dört tip bağlantı aygıtı kullanılır: tekrarlayıcı (repeater), köprü (bridge), yönlendirici (router) ve geçityolu (gateway). Tekrarlayıcılar tamamen protokol bağımsız olarak fiziksel katmanda çalışır ve fiziksel genişleme amaçlı kullanılırlar. Geleneksel köprüler ayni protokolü kullanan Yerel Ağlar arasında temel veri düzeyinde bağlantı sağlar. Buna karşılık, geleneksel yönlendiriciler değişik tipteki ağ protokollerini idare edebilecek şekilde programlanabilirler ve böylelikle aynı geniş ağ alanı üzerinde farklı tipteki Yerel Ağları ve bilgisayar sistemlerini destekleyebilirler. Geçityolları daha karmaşık olup, işlem yoğunluklu protokol çevrimi yaparak uygulamalar arasında işletilebilirliği sağlarlar.[15]

3.3.1.3.1 Tekrarlayıcı (Repeater)

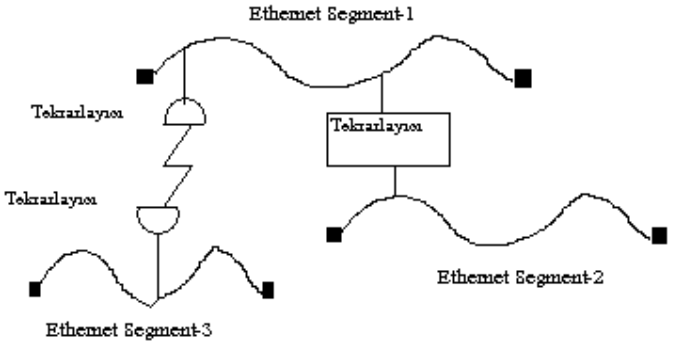
Tekrarlayıcılar şekil 3.2'den de görüleceği gibi fiziksel katmanda çalışan cihazlardır.



Şekil 3.2: Tekrarlayıcı ve OSI modeli

Tekrarlayıcının temel görevi bir fiziksel ortamdaki (kablo, fiber-optik, radyo dalgası vs.) sinyali alıp kuvvetlendirip bir diğer fiziksel ortama vermektir. Ağların fiziksel büyüklük sınırlarını daha da genişletmek amacı ile kullanılan bu cihazlar ile kuramsal olarak bir bilgisayar ağı sonsuza kadar genişletilebilir. Ancak çeşitli bilgisayar ağlarındaki tasarım sınırlamaları nedeni ile gerçekte bu genişleme belli

sınırlar içinde kalmaktadır. Şekil 3.3 tekrarlayıcıların bir ağ üzerinde nasıl kullanıldıklarını göstermektedir.

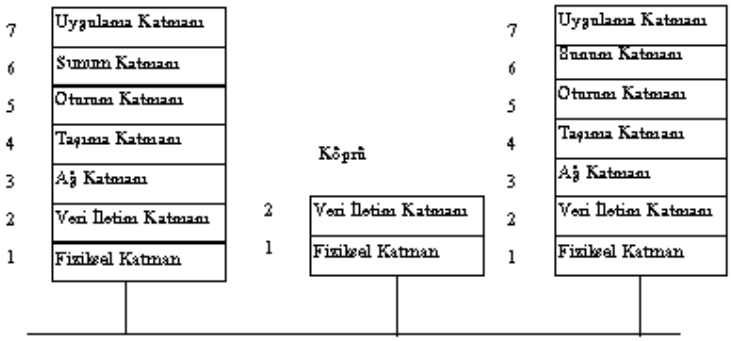


Şekil 3.3: Bir tekrarlayıcı uygulaması

Temelde bir ağın genişletilmesi amacı ile kullanılan tekrarlayıcılar çok kolay kurulmaları, çok az bakım gerektirmeleri ve fiyatlarının ucuz olması sebepleri ile çok popüler cihazlardır. [15]

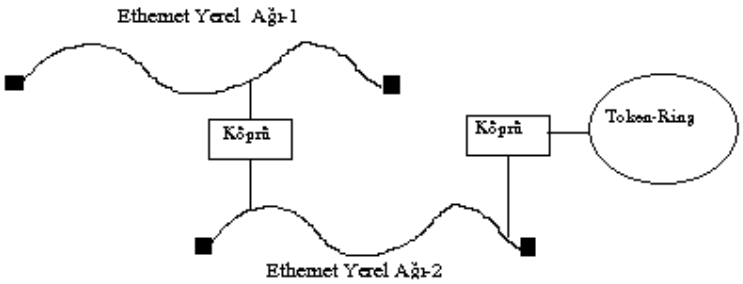
3.3.1.3.2 Köprü (Bridge)

Modern, protokol-şeffaf köprüler aşağıdaki şekilde görüldüğü gibi OSI referans modelinin veri iletim katmanında çalışırlar.



Şekil 3.4: Köprü ve OSI modeli

Köprü cihazları temelde bağımsız iki ağın, farklı ağ teknolojilerini kullanabilirler; Ethernet ve Tokan-Ring gibi birbirine bağlantısı için kullanılırlar. Şekil 3.5'de iki Ethernet ve bir Token-Ring ağının birbirlerine köprüler vasıtası ile yapılan bağlantısı gösterilmektedir. Bir köprü bağladığı alt ağlar üstündeki tüm trafiği yürütür. Her paketi okur, paketin nereden geldiğini ve nereye gittiğini görmek için MAC (Media Access Control) katman kaynağını ve yerleşim (destination) adresini inceler. Bu süzme yeteneği mesajları yayınlamak ya da yerel veri trafiğinin diğer ağ üzerine geçmesini engellemek için etkili bir yol sağlar. Bazı köprüler adres süzmenin ve protokol tipine bağlı süzgecin de ötesine gider.[15]

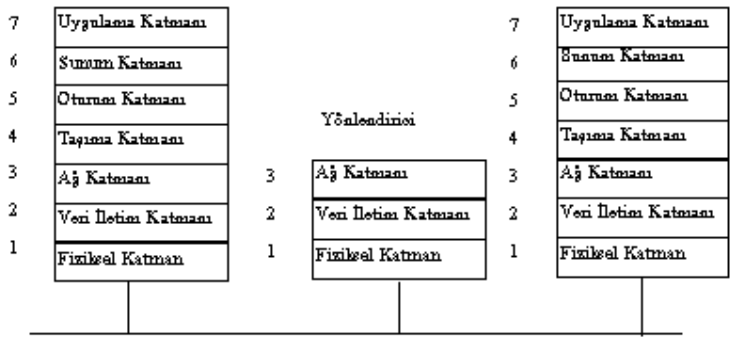


Şekil 3.5: Bir köprü uygulaması

Bir köprü, DECnet, TCP/IP, XNS gibi farklı iletişim protokollerini kullanarak, protokol uyumluluğunu göz önüne almadan ağlar arasında fiziksel bağlantı sağlayabilse de, bu uygulamalar arasında işletilebilirliğini garanti etmemektedir. Bu, OSI referans modelinin yüksek katmanlarında işleyen ve farklı işlem ortamları arasında çevrim yapabilen ek protokol çeviricilerini gerektirmektedir. Köprülü ağlar, protokol çevrimlerinin olmadığı, güvenlik gereksinimlerinin en az olduğu ve gereken tek şeyin basit yönlendirme olduğu durumlarda başarılıdır. [15]

3.3.1.3.3 Yönlendirici (Router)

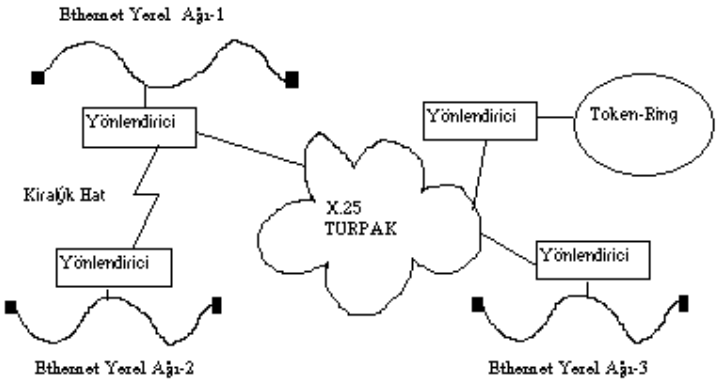
Yönlendiriciler aşağıdaki şekilde görüldüğü gibi OSI referans modelinin ağ katmanında çalışırlar.



Şekil 3.6: Yönlendirici ve OSI modeli

Bir köprü sadece paketlerin kaynağını ve gittiği yerin adresini kontrol ederken bir yönlendirici çok daha fazlasını yapar. Bir yönlendirici ağın tüm haritasını tutar ve paketin gittiği yere en iyi yolu belirleyebilmek için tüm yolların durumunu inceler. Yönlendirici farklı fiziksel yapıda olan ve farklı protokolleri çalıştıran yerel ya da geniş alan ağlarının birbirleri ile olan bağlantısında başarı ile kullanılabilir.

Bir yönlendirici, OSI referans modelinin ağ katmanında genel olarak tanımlanan protokollerle, yerel bölge ağlarını geniş bölge ağlarına bağlar. Bu özellikleri sayesinde örneğin yönlendirici TCP/IP kullanarak bir Ethernet ağının X.25 paket ağına bağlamasını sağlar. Eski yönlendiriciler protokol bağımlı olduklarından, kuruluşların ağ işletim ihtiyaçlarını karşılamak için birden fazla yönlendirici gerekebilir. Yeni yönlendiriciler ise, birden fazla ve değişik protokolü ayni anda idare edebilmektedirler.

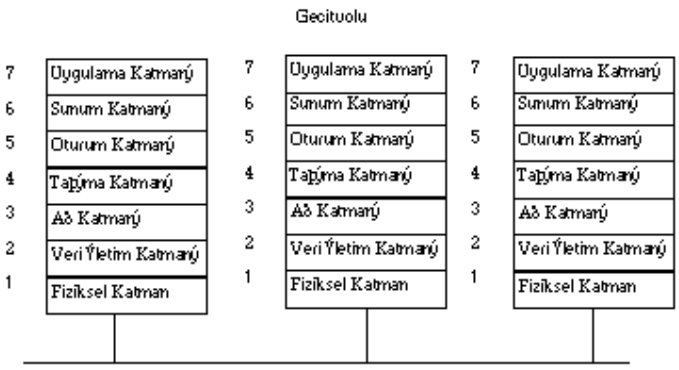


Şekil 3.7: Bir yönlendirici uygulaması

Yönlendiriciler paketleri iki istasyon arasındaki en iyi yolu gösteren yönlendirme tablosuna göre ilerleterek ağ üzerindeki yolları en iyi şekilde kullanırlar. yönlendiriciler kendi yönlendirme tablolarını oluşturduklarından, ağ trafiğindeki değişikliklere hemen ayak uydururlar ve böylelikle veri yükünü dengelerler. Ayni zamanda, yönlendiriciler ağdaki değişiklikleri tespit ederler ve aşırı yüklü ve islemeyen bağlantıları önlerler. [15]

3.3.1.3.4 Geçityollari (Gateway)

Geçityolları köprü ve yönlendiricilerin yeteneklerinin de ötesine geçerler. Aşağıdaki şekilden de görülebileceği gibi OSI referans modelinin üst katmanlarında işlerler.



Şekil 3.8: Geçityolu ve OSI modeli

Geçityolları sadece farklı noktalardaki ağları bağlamakla kalmaz ayni zamanda bir ağdan taşınan verinin diğer ağlarla uyumlu olmasını da garanti ederler. Bu bir server'da, minibilgisayarda ya da ana bilgisayarda bulunan protokol çevirim yazılımıyla yapılır. Internet protokolleri farklı ağlar arasındaki veri iletimini, geçityollarıyla bağlı alt ağlardan oluşmuş otonom sistem (Autonomous System, AS) gruplarını birbirine bağlayarak yapar. Yani Internet, her biri merkezi olarak yönetilen ağ ya da Altanlar serisi olan AS serisinden oluşmaktadır. Her AS diğer AS'lere bağlantı sağlayan geçityolu sunar. Geçityollari tüm farklı ağları birlikte tutan bir yapıştırıcıdır. Internet protokolleri altağlarin nasıl birbirine bağlı olduğunu ve bağlantı araçlarının nasıl çalıştığını tanımlar. [15]

3.3.1.4 Tcp/Ip protokol grubu

Günümüzde, heterojen (farklı topoloji ve protokollere sahip) bilgisayar ağlarını birbirine bağlamada en popüler protokoller serisi TCP/IP protokolleridir. Bu protokollerden en çok kullanılan protokol çifti ise TCP (Transmission Control Protocol) ve IP (Internet Protocol)'dir. TCP/IP grubu protokoller, Internet Protocol grubu olarak da isimlendirilir. Ancak bu bazen bilinen Internet ağı ile karıştırıldığından bu bölümde sürekli TCP/IP isimlendirmesi kullanılacaktır. TCP/IP protokol grubu birçok protokolden oluşmuştur. Burada bu protokoller sırasıyla açıklanacaktır. [14]

Bu başlık altında;

- Internet protokolünü destekleyen ve geliştiren temel kuruluşlar
- Temel Internet protokolleri tarafından sağlanan servisler
- Temel Internet protokollerinin karakteristikleri
- IP ve TCP protokollerinin alan tanımları ve işlevleri anlatılacaktır.

3.3.1.4.1 TCP/IP Protokol grubu'nun tarihçesi

TCP/IP protokol grubu, 1970'lerin ortasında, Stanford Üniversitesi ve Bolt Beranek ve Newman (BB&N) tarafından geliştirilmiştir. Geliştirme DoD (Department of Defence)'un Advanced Research Projects Agency (DARPA) bölümü tarafından desteklenmiştir. DARPA, ARPANET (Advanced Research Projects Agency NETwork) adı verilen devlet kuruluşları, üniversiteler ve araştırma kurumları paket anahtarlamalı ağlarla birbirine bağlama projesi üzerinde çalışmıştır. TCP/IP protokol grubu bu amaca yönelik olarak geliştirilmiştir. 1978-1979'larda TCP/IP protokol grubunun büyük bir kısmı tamamlanmış ve DARPA, 1980'lerde Internet protokolünü ARPANET birimlerine yüklemeye ve kullanmaya başlamıştır. 1983 yılının Ocak ayında, DARPA, ARPANET'e bağlanan tüm ağların Internet kullanmasını zorunlu tutmuştur. Internet'in büyümesi ve kullanımı ile ARPANET, küçük paket-anahtarlamalı ağlardan, noktadan-noktaya telefon bağlantılarıyla melez (hybrid) ağlara dönüşmüştür. ARPANET terimi kullanılmaya devam etmektedir ve

DoD'un araştırma ve geliştirme amacı ile Internet'in bir parçası olarak uygulanmaktadır. Internet Activities Board (IAB) adındaki bir organizasyon, şu anda Internet araştırmalarını organize etmektedir. IAB, DARPA tarafından kurulan ve Internet araştırmalarını teşvik etmeye yönelik bir kuruluştur. Her IAB grubu, Internet konularının bir parçası üzerinde çalışır. Bu çalışmaların sonuçları, çoğunlukla Internet'in işlevsel bir parçası haline gelir. Şu anda Internet üzerinde çalışan birçok protokol ve uygulama, RFC (Request For Commands) adı verilen bir dizi makale ile belgelenir. RFC kitaplığının bakımını ve jüriliğini yapma görevi, Menio Park, California'da bulunan SRI Network Information Center (NIC) tarafından yürütülür.Internet protokolünü konu alan her dökümanda, Unix BSD (Berkeley Software Distribution) ve Internet protokol birleşmesinin önemi vurgulanmaktadır. 1982'de, Unix BSD işletim sistemi üniversitelerin bilgisayar bölümlerinde çok popüler olan bir işletim sistemiydi. Ağ standardı olarak Internet'i kabul eden bu işletim sistemi ve Internet birleşimi, her ikisinin de popülaritesini artırmış ve bu durum günümüze kadar devam etmiştir. Birleşik Devletler, kendisinin denetimi altında bulunan Internet protokolü parçasının OSI referans modeli ile uyumlu olması için GOSIP (Goverment Open System Interconnection Profile) ile değiştirilmesini istemiştir. Buna rağmen TCP'IP'nin ticari kullanımı büyüyerek devam etmiştir.[14]

3.3.1.4.2 TCP/IP kullanım nedenleri

TCP/IP protokolleri belirli hedeflerin gerçekleştirilebilmesi için geliştirilmişlerdir. Bu hedefleri oluşturan talepler şunlardır. Üreticiden bağımsız tüm üreticilerin ürünlerini içine alan bir kapsam dahilinde, sistemleri birbiriyle görüştürme (IBM, DEC, Sun, HP vb.)

- Tüm ölçekteki bilgisayarları birbirleriyle görüştürme (PC, Midrange Systems, Mainframe vb.)
- UNIX sistemlerle tam uyumluluk.
- Dinamik router teknolojisinin desteklenmesi.
- İstemci/Sunucu bilgi işleme teknolojisinin desteklenmesi.
- Bir-e-Bir(Peer-tp-Peer) yapılanmasına uygun teknolojiye sahip olması. [14]

3.3.1.4.3 TCP/IP'nin Katmanları

Ağ arayüz katmanı (Network Interface Layer)

DoD modelinin en alt katmanıdır. Fiziksel iletişim ortamı üzerinden frameleri hedefe gönderen katmandır. OSI modelinde, Fiziksel katman ile Veri-Bağlantı katmanının her ikisine birden karşılık gelir. Bu katmanın TC/IP protokol kümesinden herhangi bir tanımlaması yapılmamıştır. Bu yönüyle varolan bütün Fiziksel ve Veri-Bağlantı katmanı protokollerini destekler. [14]

Internet katmanı (Internet Layer)

DoD modelinin 2. katmanını oluşturur. OSI modelinde Ağ katmanına karşılık gelir.Host'dan Host'a haberleşmenin sağlanmasından sorumludur. Yol belirleme algoritmaları (Distance-Vector, Link-State), çalışmalarında bu katmanı da kullanırlar. [14]

Aktarım katmanı (Transport Layer)

DoD modelinin 3. katmanıdır. Farklı hostlar üzerindeki uygulamaların birbirleriyle görüştürülmesinden sorumludur. Datagram paketleri içerisinde kimlik bilgileri burada yerleştirilir ya da çözülür (soket numaraları). Aktarım katmanı karşılıklı işlem bazında (Process-to-process) görüşme sağlar. TCP/IP protokol grubundan bağlantısız servis veren UDP ile bağlantı tabanlı servis veren TCP, bu katmanının iki protokolüdür. [14]

Uygulama katmanı (Application Layer)

DoD modelinin en üst katmanını oluşturur. HTTP, TELNET, FTP, SMTP ve SNMP gibi TCP/IP protokolleri bu katman çalışır. Datagram bir hosttan başka bir hosta iletilmek üzere oluşturulmuş veri gruplarına verilen isimdir. Genelde Aktarım

Katmanındaki veriye Datagram denir. Ancak bazen UDP ve TCP verilerini ayırt etmek için TCP'nin verisi için "segment" ismi kullanılır.[14]

3.3.1.4.4 TCP/IP www hizmeti

World Wide Web (WWW)

WWW yada W3 yazı, resim, ses, film, animasyon gibi çok farklı yapıdaki verilere kompakt ve etkileşimli bir şekilde ulaşılmasını sağlayan çoklu hyper ortam sistemidir.Hiper ortam bir dökümandan başka bir dökümanın çağrılmasına (navigate;-iç içe dökümanlar-) imkan sağlar. Bu ortamdaki her veri bir başka veriyi çağırabilir (link). Link aynı doküman içinde başka bir yere olabildiği gibi, fiziksel olarak başka bir yerde de (internet üzerindeki herhangi bir makinede) olabilir.Bütün bu farklı yapıdaki veriler uygun bir standart ile bir arada kullanılıp bir Web listeleyicisi'nde (web browser) görüntülenebilir. Web'in diğer bir işlevi de, özeki bazı internet servislerini kendi içerisinde barındırmasıdır. (ftp, gopher, news, wais) Web uygulamaları (web sayfaları) web listeleyicilerinde (browser, gezgin, tarayıcı) görüntülenebilir. Web sayfaları, başka sayfalara ve değişik türden verilere hiper linkler içermektedir. Buralara fare ile tıklayarak başka sayfalara, oradan da başka sayfalara geçilir. Bu aslında çok basit bir bilgiye erişim modelidir. Web sistemleri, kullanılan platformdan bağımsızdır. Bir Macintosh, PC ya da Unix Web Listeleyicisi aynı sayfaları, aynı şekilde alırlar. Sayfaların alındığı Web Servisleri de farklı bilgisayar platformlarında olabilir. Web Listeleyicileri ve Web Servis Sağlayıcı ortamlar dünyada her yerde vardır ve global olarak kullanımları üstel bir şekilde artmaktadır. Web yapısının bu kadar çok kabul görmesinin temel nedenleri aşağıdaki gibidir.

- Web açık bir sistemdir; platform, bilgisayar ve işletim sistemi gibi bağımlı değildir
- Web üzerinden pek çok bilgi kaynağına kolaylıkla erişilmektedir.
- Web uygulamaları geliştirmek ve bunları kullanıma sunmak kolay bir işlemdir. Çoğu durumlarda, fazla bilgisi olmayan kullanıcıların bile web

sayfası dizayn edip kullanıma sunması uzmanlık gerektirmemesi nedeniyle mümkündür.

- Web ortamları artık son derece dinamiktir. Java ve ActiveX kullanarak, tamamen konfigüre edilebilir. İstemci (Client) uygulamaları geliştirerek, java (web sayfalarının içerik ve görselliğini zenginleştirmede kullanılan bir programlama dilidir. Java Scripts ve Java Applet gibi türleri vardır.) uygulamalarıyla firma, ürün tanıtımları yapılabilir.Bir kullanıcı, isteğe bağlı olarak, bağlandığı bir veri tabanından istediği bilgileri farklı gruplarda isteyebilir. (client side corparation)
- Aranılan bilgilere, bir takım tarama mekanizmaları (search engines) sayesinde kolayca ulaşılabilir. [14]

3.3.1.5 İnternette isimler ve adresler

3.3.1.5.1 Adresleme stratejileri

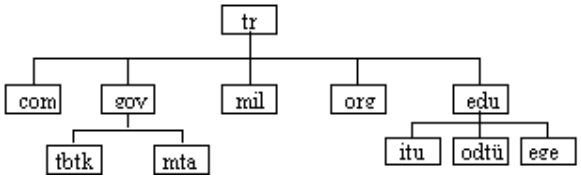
İlk bilgisayar sistemleri, kullanıcıların sayısal adresleri anlamaları ve kullanmaları temelinde tasarlanmışlardı (sistem tabloları, yazıcı ve teyp üniteleri gibi cihazlar vs.). Daha sonra ortaya çıkan sistemlerde harici cihazlar (yazıcı vs.) ve dosyalar daha anlaşılır sembolik isimler ile gösterilmeye başladı. Benzer bir değişiklikte ağ bağlantılarında yasandı. Önce bilgisayarlar arası noktadan noktaya bağlı ağ teknolojisi ortaya çıktı ve alt seviye donanım isimleri makineleri tanımlamada kullanıldı. Ancak pek çok bilgisayarın birbiri ile bağlantısı gündeme geldiğinde üst seviye adresleme yapısına gereksinim doğdu. Kullanıcılar pek çok makineden oluşan hesaplama ortamlarında makineleri tanımlamak için anlaşılır sembolik isimlere sahip bir adresleme yapısını talep ettiler. Bilgisayar sayılarının günümüze göre çok az olması sebebi ile başlangıçta sadece makinenin kullanım amacına yönelik bir adlandırma yöntemi kullanıldı (personel, araştırma, muhasebe, geliştirme vs.). Ancak makine sayısının artması ile sembolik yeni isimlerin bulunması ve tüm bu birbirine bağlı sistemlerin adlarının bir merkezden kontrolü zorlaşmaya başladı. [15]

Adresleme problemlerini en aza indirmek için, merkezi olarak bilgisayar isimlerinin kontrolü ve kaydı yerine daha uygun bir sistem olarak sıradüzensel (hiyerarşik) ve otoritenin dağıtıldığı merkeziyetçi olmayan bir adresleme sistemi getirildi. Bu sistemde adresleme en genelden özele doğru yapılmakta ve her adres seviyesinin kontrolü yetkisi de dağıtılmaktadır. Bu yapıya 'Alan İsimlendirme Sistemi-Domain Name Sistem' veya kısaca DNS ismi verilmektedir. Hiyerarşik yapıdaki Alan (Domain) isimleri kavramını biraz daha detaylı inceleyelim. [15]

Alan (Domain) ismi birbirinden bir nokta (.) ile ayrılan, sıradüzensel seviyedeki alt isimler (subnames) dizisidir. Mesela ODTU Bilgisayar Merkezi Alan ismi olan bilmer.selcuk.edu.tr dört seviye ile gösterilir ve her bir seviyeye de Domain adi verilir. Örneğimizde en alt seviye olan 'bilmer' Bilgisayar Merkezini göstermektedir. Üçüncü seviye 'selcuk', Selçuk ‘un Domain ismidir. Bir üst seviye 'edu' (Education) ise bu domain'in bir eğitim kurumuna ait olduğunu gösterir. En üst seviye 'tr' ise ISO (Internetional Standards Organization) tarafından belirlenen Türkiye’nin ülke kodudur. En üst seviyede kullanılan bazı domain isimleri aşağıda listelenmiştir:

.com	ticari kuruluşlar (commercial)
.edu	eğitim kuruluşları (education)
.gov	devlet kuruluşları (government)
.mil	askeri kuruluşlar (military)
.net	ağ organizasyonları (network)
.ulke kodu	ISO standart ulke kodu

Kısacası isimler aşağıdaki domain ve alt domain sıradüzensel yapıya göre verilir:
Makine.altorganizasyon.organizasyon.domain



Şekil 3.9 : İsimlendirme organizasyonu

3.3.1.5.2 TCP/IP ve DNS

Bilindiği gibi TCP/IP ağlarına bağlı olan her bilgisayarın ağ arayüzü 32-bitlik IP adresi ile tanımlanmaktadır. Ancak IP adreslerinin gündelik hayatta kullanımı ve hatırlanması pek pratik olmadığı için domain isimlendirme sistemi kullanılır. Aslında TCP/IP yazılımlarının ağ üzerindeki iletişimi sağlamak için isimlere ihtiyacı yoktur, isim yapısı ağ kullanıcılarının hayatlarını kolaylaştırmak için ortaya çıkarılan bir yöntemdir. Kullanıcının tercihine göre IP numaraları veya isimler kullanılabilir.
Mesela:
% telnet 144.122.199.20
% telnet knidos.cc.metu.edu.tr
komutlarının her ikisi de ayni işlevselliktedir. Her iki durumda da bağlantı IP numarası kullanılarak yapılır. İsim ile bağlantı durumunda sistem önce bilgisayar ismini (knidos.cc.metu.edu.tr) IP numarasına çevirir ve daha sonra bu numaraya bağlantıyı sağlar. Dikkat edilirse bu sistem sayesinde makinenin IP adres değişiklikleri kullanıcıyı hiç etkilememektedir. [15]

İsimler ve adresler arasındaki ilişkiyi sağlayan sistemleri ve programları kurup çalıştırmak sistem sorumlularının görevidir. İsimlerin IP adreslerine çevrilmesi işlemi aslında çok basit bir işlem de değildir. Zira lokal çalışan bir bilgisayar ağında hiç dikkat edilmeyen pek çok konu bilgisayar ağını İnternete bağladığınızda çok ciddi problemlere yol açabilir. Ağ üzerinde yer alan bilgisayarlarınızın isminin IP adresine çevrilmesi artik dünyanın her yerinden sorunsuz olarak yapılmak zorundadır. [15]

DNS in nasıl çalıştığını ticari bir şirketin İnternete bağlandığını varsayarak bu örnek üzerinde açıklamaya çalısalım. Bu hayali şirketimizin adi PLANET AS. olsun. Şirketimiz su anda Türkiye de çalıştığı için tabiî ki üst seviye domain adi .tr olacak. Alt seviyede ise şirket olmasından dolayı .com domaini altında bulunmaktadır. Bir alt domain ise şirketimizin adini göstermektedir, planet.com.tr. Büyük bir şirket olduğumuz için farklı birimlere sahibiz ve her birimimizde ayrı bir domain altındadır. Araştırma geliştirme bolumu arge.planet.com.tr, satış bolumu sales.planet.com.tr, destek bolumu support.planet.com.tr gibi. [15]

Birimlerimiz oldukça büyük olduğu için her birim kendi Domain Name Servisini kendisi kontrol etmektedir. Şirketimizin ayrıca tüm bu alt seviye domainleri tanıyan bir "root name server" makinesi bulunmaktadır. Alt domainler sadece kendi domainleri ile ilgili bilgiyi ellerinde tutarlar ve bilemedikleri her turlu domain için sorgulamayı "root name server" üzerinden yaparlar. Ayrıca eğer istenirse alt seviye domainler (.sales, .arge gibi) kendi içlerinde başka alt seviye domainler de (sub domain) yaratabilirler. Aslında tüm seviyelerdeki domainleri kontrol eden "name server" makineleri kendi sorumlulukları altındaki bilgisayarların isimlerini ve IP adreslerini tablolarda tutan birer Bilgi Bankasından (Database) başka bir şey değildir.

Şirketimiz bu yapıyı kurduktan sonra doğal olarak Internet üzerindeki başka merkezlerle alfanumerik adresler kullanarak haberleşmek isteyecek ya da dışardan kullanıcılar şirketimizin sunduğu bazı servislerimizi alabilmek için bize ulaşmak isteyeceklerdir. Bu noktada şirketimizin "root name server" makinesini ülke içindeki root name server makinesine tanıtmamız gerekmektedir. Su anda Türkiye içindeki .com dahil tüm domainler için bu görevi knidos.cc.metu.edu.tr adresinde bulunan bir UNIX makine yapmaktadır. Bu "name server" üzerinde PLANET şirketinin root name server kaydı yapıldıktan sonra artik dünyanın dört bir yanına alfanumerik isimler kullanarak ulaşmak için hazırız demektir.[15]

Şimdi Amerika'daki bir kullanıcının şirketimizin satış bölümündeki unix1.sales.planet.com.tr isimli bilgisayara Internet üzerinden ulaşmak istediğini varsayalım.

Şekil 3.10 : İsimlendirme örneği

Şekilde de görüldüğü gibi unix1.sales.planet.com.tr adresine ulaşmak için lokal server önce ABD'de icm1.icap.net adresindeki name server'a sorguyu yolluyor.

icm1.icap.net Türkiye ile ilgili bütün kayıtların knidos.cc.metu.edu.tr adresinden alınacağını bildiği için sorgulamanın bu adresten yapılmasını istiyor. Ayni sorgu bu sefer knidos.cc.metu.edu.tr adresine yollandığında sorgulanan adresin netmgr.planet.com.tr tarafından bilindiği cevabi yollanıyor. Ve sonuçta ulaşılmak istenen adres (193.141.75.9) netmgr.planet.com.tr adresinden elde ediliyor. [15]

Bu sorgulama sonucu Amerikandaki kullanıcının makinesi unix1.sales.planet.com.tr makinesinin IP adresini öğrenmiş oldu ve bu adres ile yapmak istediği iletişimi sağladı. Bu sorgulamanın sonucu ayrıca istekte bulunan bilgisayarın cache belleğine yerleştirildi. Bu bilgi cache bellekte durduğu surece bir daha ayni adrese bağlanmak isteyen bir kişi tekrar ayni sorgulamayı yapmaksızın o IP adresine doğrudan ulaşabilecektir. [15]

3.3.2 Web Programcılığı

Web programcılığını statik ve dinamik programlama olarak ikiye ayırabiliriz. Statik ve dinamik programcılık birbirinden farklı kavramlar olmasına rağmen çoğu zaman birlikte bir bütün oluşturmaktadırlar.

Web programcılığı kavramının ilk aşamasında ziyaretçiler web sayfalarını okumak, izlemek, takip etmek amacıyla ziyaret ediyorlardı. Ziyaretçiler gazete okur gibi sabit sayfaları geziyor web ortamına müdahale edemiyorlardı. Örneğin sayfalara kendi bilgilerini gönderemiyor mantıksal sorgulamalar yapamıyor interneti tam anlamıyla kişiselleştiremiyorlardı. Bu sayfalar HTML denilen web sayfası işaretleme dili kullanılarak geliştirildiler. HTML ile oluşturulan web sayfaları yapısı gereği web sunucular tarafından hiçbir işlemden geçmeksizin eğer sayfada kullanılmışsa ses, resim, animasyonlarla birlikte direk sabit diskten okunarak ziyaretçilerin bilgisayarlarına gönderilirler. Bu şekilde yapılan web programcılığı günümüzde statik web programcılığı adını almaktadır. Günümüzde ihtiyaçlar doğrultusunda hala kullanım alanı bulmaktadır ve çoğu HTML'e gömülü çalıştığı için dinamik programcılığın kodlarını oluşturma da kullanılmaktadır.

Dinamik programcılık ise HTML dilinin yetersiz kalması ve ziyaretçilerin beklentilerin artmasıyla geliştirilmiş ve kullanılmaya başlanmıştır. Bu kavramla birlikte sayfalarda anketler, formlar, forumlar, mesajlaşma, kişileştirilmiş web sayfaları, yönetim arayüzleri oluşturulmuş ve kullanılmıştır. Dinamik programcılığın en can alıcı özelliği ise web sayfalarının veri tabanlarına bağlanmakları için prosedürler oluşturmuş olmasıdır. Dinamik web programcılığı dillerine örnek olarak ASP, ASP.NET, PHP, JSP, JAVA, Javascript, Vbscript gösterilebilir. Statik programcılıkta ise HTML ve XML dilleri kullanılmaktadır.

3.3.2.1 HTML (Hypertext markup language)

Hyper Text Markup Language (hareketli metin işaretleme dili) baş harflerinin kısaltmasından oluşan HTML ile yazı, resim, ses, film gibi pek çok farklı özellikteki dosyaları formatlayarak, boyutlarını ve görünecekleri yerlerini ayarlayıp gösterilmesi sağlanabilir. Ayrıca HTML köprü denilen özelliği ile internette bulunan başka bir sayfaya yönlendirme yapılabilinir.

Bir HTML dökümanının genel olarak görünümü aşağıdaki gibidir:

<HTML> Html programının başlangıç komutudur.

<HEAD>

Bu alanda yazılan bilgiler web sayfasında görüntülenmezler. Burada sayfa başlığı, anahtar kelime tanımlamaları, sayfa içerisinde kullanılan karakter bilgisi (dil, code page) gibi sayfanın doğrudan kendisine ilişkin tanımlamalar yapılır.

</HEAD>

<BODY>

Sayfa üzerinde görülmesi istenen herşey bu bölümde yazılır. Hazırlanan dokümanın başlangıç ve bitiş bloğu gibidir.

</BODY>

</HTML> Html programının bitiş komutudur.

Yukarıda da görüldüğü gibi HTML komutları (belirteçleri) < > işaretleri arasına yazılırlar ve genelde her HTML belirtecinin / ile başlayan bir çifti vardır. Belirteçlerin çiftler halinde bulunması; söz konusu belirtecin sağladığı özelliğin sadece belirteç çifti arasına yazılan yazılara etki edeceğini ifade eder. Bir belirtece ait birden çok seçenek bulunabilir ve belirteçler seçeneklerle kullanıldığı zaman, bu seçeneklerin bir de değeri bulunur.

<belirtec_adi secenek_adi1=deger_1 secenek_2=deger_2 ...>

<font size=4>

HTML, büyük harf küçük harf duyarlılığı olmayan bir dildir. HTML ile oluşturulan web sayfaları metin tabanlı oldukları için platform bağımsız çalışmaktadırlar. Bu ise HTML' e büyük güç katmaktadır.

HTML in sağladığı komut ve özelliklerle çok seviyeli başlıklar, paragraflar, hipermetin referansları, maddelenmiş listeler için özel formatlama, satır içi görüntüler ve doldurulabilen form, sayfa görünümü üzerinde ileri derecede kontrol, manşetler, görüntülerdeki popüler noktaların istemci tarafında işlenmesi, özelleştirilmiş listeler, stil yaprakları, form içi tablolar yapılar sağlanabilir.

Kısaca HTML sadece gösterim dili olarak düşünüldüğü için görüntüleme stilleri yaratmak için bir takım özellikler sunmaktadır.

3.3.2.2 XML (Extensible Markup Language)

XML (eXtensible Markup Language) genişleyebilir işaretleme dilleri anlamı gelen kelimelerin baş harflerinden oluşmaktadır. XML daha çok iki veya daha fazla uygulama arasında veri alış verişi için ortak bir dil kullanma ihtiyacı neticesinde oluşturulmuş bir dildir. XML genişleyebilir, birlikte çalışabilir ve istediğimiz etiketi tanımamıza imkan sağlayarak özelleştirilebilir. İşletim sistemi uygulama platform bağımsız olması metin tabanlı yazılmasının bir sonucu olarak düşünülebilir.

XML, HTML in eksikliği gidermek için oluşturulmuş ve yapısal veri sağlamak amacına hizmet etmektedir. Temel yapı hiyerarşik etiketlere dayanır. Metin tabanlı etiketler alt alta sıradüzensel olarak bir araya gelir ve olan bir dosya olarak kaydedilir.

XML bir meta dildir. Diğer bir değişle diğer işaretleme dillerini tanımlamak için kullanılan bir dildir XML ile herhangi bir uygulama için XML belgesinin içinde bulunacak verinin içeriği ve içerdiği veri tiplerini tanımlayacak uygulamaya özel bir işaretleme dili tanımlayabilirsiniz.

Örnek bir XML uygulaması olarak

```
<?xml version='1.0'?>
<HAVADURUMU>
<SEHIR>
<ADI>KONYA</ADI>
<HAVA>ACIK</HAVA>
<SICAKLIK>35</SICAKLIK>


</SEHIR>
<SEHIR>
<ADI>MANISA</ADI>
```

```
<HAVA>KAPALI</HAVA>
<SICAKLIK>20</SICAKLIK>
</SEHIR>
</HAVADURUMU>
```

Örnekteki kodlar “havadurumu.xml” dosyası olarak kaydedilirse bu dosya işlenirken şehirleri ve beklenen hava durumu tahminlerini çok kolay bir şekilde alabiliriz.

3.3.2.2 Javascript

JavaScript, Java dilinin bir kırpılmış versiyonu değildir. JavaScript kendi başına bir script dilidir. Genellikle HTML sayfalarında kullanıcı kontrolleri eklemek, formları kontrol etmek, hesaplamalar yapmak gibi işler için kullanılır.

JavaScript kullanımının iki ana alanı vardır: istemci taraflı ve sunucu taraflı script yazmak. Şu anda HTML sayfalara gömülü yazılan scriptlerin çoğu JavaScript kullanılarak yazılmaktadır. Bunlar da genellikle kullanıcı bilgilerinin doğru girilip girilmediği ya da kullanıcıya veri girişinde ya da menü kullanımında seçenekler sağlamaktır. Sunucu taraflı JavaScript (SSJS) ise temel JavaScript' e ek olarak nesneler ve işlevler katılarak veritabanlarına erişim, e-mail gönderme ve diğer işlemlerin yapılması sağlanır. SSJS, veritabanı temelli web uygulamalarının yaratılmasını sağlar.

Bir tarayıcının anlayacağı temel dil HTML' dir. JavaScript dilinin HTML belgesi içinde yazılabilmesi için <SCRIPT> etiketleri kullanılır.

```
 <HTML>
 <SCRIPT>
...
//JavaScript kodu
//Örnek
document.write ("Merhaba");
```

....
</SCRIPT>
</HTML>

JavaScript dilinin temel gramer yapısına bakılırsa, HTML belgesinde JavaScript eklendiğinde, kodun işletimi sırasıyla yapılır. Ancak fonksiyon ve belli bir olaya (ONCLICK gibi) bağlı olan JavaScript kodlarının işletimi o anda yapılır. Değerler ya da değişkenler arasında toplama, çıkarma gibi işlemlerin yapılmasını sağlarlar. İşleçler +, *, /, - gibi işaretlerle kullanılır. Değişkenlerin, işleçleri ve deyimlerin bir araya gelerek oluşturdukları yapılara ifade denir. Örneğin: "Ucret = Gün * Yevmiye" formülü bir ifadedir. Deyimler belli bir komut ya da söz dizimi bileşenlerinin grubuna verilen addır.

Örneğin bir komut ya da bir IF yapısı deyimi oluşturur:
if (toplam>20) {deyimler;} else {deyimler;}

Kendi değerleri, özellikleri ve işlemleri olan bileşenlere nesne (object) denir. JavaScript dilinde çok sayıda yerleşik nesne vardır. Bir JavaScript fonksiyonu diğer dillerdeki bir fonksiyon yordam ya da bir alt yordam anlamına gelir. Bir fonksiyon bir dizi işlemi yerine getirir ve bir sonuç döndürür. Ayrıca fonksiyonlar, parametre olarak kabul edilen birçok değer kabul ederler. JavaScript büyük küçük harf duyarlı bir dildir. Değişkenler ve diğer öğeler küçük ya da büyük yazılabilirler. Ancak A değişkeni ile a değişkeni birbirinden farklıdır.

3.3.2.2 CGI (Common Gateway Interface)

CGI kelimesi, Common Gatevay Interface kelimelerinin başharflerinden oluşan bir kısaltmadır. Türkcesi de "ortak geçit arayüzü" anlamına gelir. CGI ile ziyaretçi tarafından server'a bir iş yaptırılır. Bu iş ziyaretçinin veritabanı olarak kullanılan dosyalara belli konularda kayıt yapabilmesi şeklinde olabileceği gibi ziyaretçi sayısının sayılması, web üzerinden mail gönderilmesi şeklinde geniş bir yelpazede devam eder. CGI programlarında c++, visual basic scripting gibi diller

kullanılabilse de genel olarak CGI programlarında perl dili kullanılır. CGI programlarının işleyişi hakkında örnek verecek olursak web sitemizi ziyaret eden kişilerin görüşlerini bildirebileceği bir ziyaretçi defteri yaptığımızı düşünelim. Web sitemize gelen ziyaretçi, ziyaretçi defterine girer, görüşlerini bir form vasıtasıyla doldurur ve Gönder basarak formun "action" kodlarıyla belirlenen ve form vasıtasıyla bilgilerin gönderileceği adres olan CGI ya da PL dosyasına bilgiler "bilgi yumağı" halinde gönderilir. bilgileri alan CGI dosyası bu bilgileri kendisine verilen komutlar doğrultusunda açar, değişkenlere atar, dosyaya kaydeder ve isteğe bağlı olarak mail vasıtasıyla defter sahibinin mail adresine gönderir. Sonuç olarak da ziyaretçiye bir teşekkür içeren html dosyası gösterir. Örneğimizde ziyaretçi CGI dosyasına bilgileri gönderdi ve CGI dosyası da server'a bir "iş" yaptırdı dosya açtırdı, dosyaya kaydetti, dosyayı kapattı, mail gönderdi ve teşekkür mesajı içeren html dosyası yazdırdı.

Özellikle Java ve JavaScript kullanımlarının yaygınlaşması, CGI'nin ilk zamanlardaki popülaritesini azaltmıştır. Öte yandan, CGI programları/scriptleri, bulundukları sistemde saklıdırlar ve çalıştıklarında o sistemin kaynaklarını kullanırlar. Oysa Java Appletleri ve JavaScript, doğrudan HTML döküman içinde onu çağıran web listeleyicisine gelirler ve burada icra edilebilirler. Bu da, Java Applet ve JavaScript'in CGI’ ye göre önemli bir avantajıdır.

Cgi programları bildiğimiz düzyazı dosyalarında saklanır ve uzantıları. pl veya .cgi olarak kaydedilir. Tarayıcıdan bu dosya çağırıldığında, web server bunun bir cgi dosyası olduğunu anlar ve yorumlayıcı satırında belirtilen yorumlayıcıya yorumlaması için gönderir. Yorumlayıcı programın dediklerini yapar ve sonucu tarayıcıya gönderir.

Örneğin 2+3=5 işlemini yapan bir program şu şekilde olabilir;

```
#!/usr/bin/perl
print "Content-type:text/html\n\n";
$iki=2;
```

```
$uc=3;
$toplam=$iki+$uc;
print"2+3=$toplam eder.\n";
```

Bu kodları Windows'un Not Defteri'nde yazıp uzantısını .cgi ya da .pl şeklinde kaydetmek yeterlidir. Bu dosyayı tarayıcıdan çağırdığımızda ekrana **"2+3= 5 eder"** yazacaktır. Cgi programlarının kodları web server'da icra edildiğinden tarayıcı tarafından görüntülenmez.

3.3.2.3 ASP (Active Server Pages)

ASP dinamik web sayfaları hazırlamak için Microsoft tarafından geliştirilmiş bir teknolojidir. Web sunucudan HTML bir dosya talep edildiğinde web sunucu HTML dosyayı ve içerdiği resim, animasyon ve diğer öğeleri sabit diskten okuyarak ziyaretçinin bilgisayarına gönderir ama bu yapı ASP böyle değildir eğer web sunucuya uzantısı Asp uzantılı bir dosya talep edilmişse web sunucu bunu işlemek için asp.dll denen asp motoruna bu sayfayı incelemesi için gönderir. Asp.dll motoru sayfanın içerisinde geçen komutları,fonksiyonları varsa veritabanı bağlantısını sağlayarak geriye html kodlarından oluşan sayfayı hazırlar ve ziyaretçinin bilgisayarına gönderilir. Asp kullanıcıdan talep gelmediği sürece akıp giden bir programlama değildir. Yani Sonuç olarak kullanıcıya iletilecek html dosyası kullanıcı istekte bulunana kadar oluşturulmaz. Asp dinamik web programlama konusunda yalnız olmamasına rağmen günümüzde oldukça teknik destek ve gelişin gösteren bir dildir. CGI programla yönteminde sunucu bilgisayara her talep geldiğinde hafıza da yeni bir exe oluşturulması ile çok talep olan web sitelerinde sistem darboğazları oluşmaktaydı ASP de ASP.DLL denen ISAPI kullanılmasıyla bu durum farklıdır. ISAPI, tıpkı CGI gibi, Web Server programının bulunduğu bilgisayardaki diğer programlarla alışverişini sağlar. Ne var ki ISAPI' den yararlanan programlar üreterek bunları Web Server'ın emrine vermek oldukça pahalı bir yol. Başka bir deyişle, bir formda Gönder düğmesinin çalışabilmesi için sözgelimi Excel ayarında bir program yazdırtmak pek akıl karı olmasa gerek. Bu noktada çeşitli firmalar ISAPI benzeri "yorumlayıcılar" geliştirerek, düz yazı bir Script yazmakla ve

bu Script'in içindeki komutları Server programına icra ettirmekle, Web tasarımcısının hayatını çok kolaylaştırabileceklerini gördüler. Microsoft'un bu noktadaki çözümü ASP oldu.

ASP, bir teknolojidir. Kendi başına bir yazım kuralı yoktur. ASP tekniğini kullanabilmek için, ASP sayfasının talep edilmesi halinde ziyaretçiye gönderilmeden önce ASP.DLL'ye teslim edilmesi bu teknolojinin kullanılabilmesi için hemen hemen tek şarttır. Bunu, dosya uzantısını .asp yaparak sağlarız.
ASP.DLL ise, dünyada mevcut bütün Script dilleri ile verilecek komutları kabul edebilir. Sadece ASP.DLL'e sayfadaki kodların hangi dilde olduğunu söylemeniz gerekir. Bunu, ASP sayfasının birinci satırında yaparız. Örneğin ASP'ye VBScript dilini kullanmasını belirtmek için bu satırı şöyle yazarız:
<% @Language=VBScript %>
ASP sayfalarında genellikle VBScript, JavaScript ve JScript kullanılır. Ancak örneğin Perl dilinden türetilen PerlScript, PHP'den türetilen PHPScript de giderek ilgi çeken ASP dilleri arasına giriyor.[16]

3.3.2.4 ASP.NET (Active Server Pages)

Çıkalı henüz beş-altı yıl olan ASP (Active Server Pages) çok büyük bir beğeni topladı ve Internet uygulamalarının çoğu ASP ile geliştirildi. Bu ilginin sebebi ise ASP ile dinamik Web siteleri ve dağıtık Web uygulamalarının geliştirilebilmesidir. XML'in öneminin artması ve gelişen teknolojiye ayak uydurabilmek için Microsoft ASP'yi geliştirmeye başladı ve bu yeni versiyona ASP+ adı verildi. Microsoft, .NET stratejisine uygun olması için bu adı daha sonradan ASP.NET olarak değiştirdi.

ASP.NET, .NET Framework'ün çok önemli bir kısmını oluşturmaktadır. ASP.NET, dinamik Web uygulamalarının geliştirilebilmesi için gerekli alt yapıyı sağlamaktadır ve web uygulamalarının çok daha az zamanda geliştirilebilmesini sağlayacak şekilde dizayn edilmiştir.

ASP'de uygulama geliştirirken karşılaşılan büyük bir sorun sadece script dillerinin kullanılabilmesi ve bu dillerden hangisinin kullanılacağının doğru seçilmesidir. Internet üzerinde çalışacak bir uygulama geliştiriyorsak Jscript veya JavaScript dillerinden birini kullanmak zorundayız, çünkü VBScript sadece Internet Explorer üzerinde çalışmaktadır. Fakat Intranet üzerinde bir uygulama geliştiriyorsak VBScript kullanarak Internet Explorer Web gezginini güçlendirmek mümkündür. ASP.NET'in ASP'den en büyük üstünlüğü dinamik Web uygulamaları geliştirirken dilden bağımsız olması ve her platform ve cihaz için çıktı üretebilmesidir. ASP.NET'de birçok dil kullanma şansınız var. Hangi programlama dilini kullanıyor olursanız olun herkes aynı objelere, özelliklere ve metotlara erişebilir. COM objelerinden farkı sadece çağırmakla kalmıyorsunuz aynı zamanda farklı bir programlama dilinde yazılmış bir sınıfı kalıtım edebiliyorsunuz. [17]

ASP.NET İle Gelen Yenilikler

ASP.NET birçok açıdan eski versiyonu olan ASP'den üstündür. Aşağıda ASP.NET'in bu üstün özellikleri açıklanmıştır:

Daha Az Kod

ASP.NET'in üstün özelliklerinden biri, uygulama geliştirirken daha az kod gerektiriyor olmasıdır. Satır sayısında %40 ile %70 arasında azalma olur. Önceden HTML ve script kullanarak yapabildiğimiz işlerin birçoğunu artık sadece bir sunucu kontrolü koyarak yapabiliriz. Örneğin ASP'de bir metin kutusunun kullanıcı tarafından boş bırakılmaması ve sadece belirli aralıktaki sayısal verilerin girilmesi gerektiği bir durumda bir script yazarak o metin kutusunun içine bir şeyler yazılmış mı yazılmışsa belirli koşulları sağlayıp sağlamadığını kontrol ederdik. Oysa şimdi aynı işi sadece bir RequiredFieldValidator kontrolü koyarak yapabiliriz. Bir Calendar kontrolü yaratarak ekrana çok hoş bir takvim koyabiliriz. ASP.NET, kullanıcının durumunu otomatik olarak tutar ve böylece ASP'deki gibi bir kullanıcı sayfayı yenilediğinde kullanıcının durumunu tutmak için script yazmak zorunda kalmayız.

Uygulama ve Sunumun Ayrılması

ASP sayfaları kolay ve çabuktu fakat HTML ve onu geliştiren sunucu taraflı mantık birbirine karışıyordu. Örneğin:

<H2><%= “” & adisoyadi & “” %></H2>

“<%=” ve “%>” şeklindeki sunucu taraflı mantık ile HTML kodları içi içe geçtiği için bu tür bir kodu değiştirmek, hatalarını ayıklamak ve devamlılığını sağlamak zordur.

ASP.NET ile sunucu taraflı web form ve web servisi geliştirdiğimizde içerik ve kod birbirinden tamamen ayrılır ve farklı dosyalarda olabilir.

Derlenmiş Kod

ASP sayfaları yorumlanarak çalıştırılıyordu fakat ASP.NET sayfaları yorumlanmaz, derlenir. Bir ASP.NET sayfası ilk kez çalıştırıldığında bir .NET sınıfı içine derlenir. Bu sınıf ön bellekte depolanır ve sayfaya yapılacak olan sonraki isteklerde ön bellekten kullanılır. ASP.NET sayfasında bir değişiklik yapıldığında bu değişiklik fark edilir ve ilk istekte tekrar derlenir ve ön belleğe kaydedilir. Sayfa, her istek için yorumlanmak zorunda kalmadığından performansta büyük bir artışa neden olur.

Değişebilen Mimari

ASP’ de sunulan imkanları programcı beğenmese de aynen kullanmak zorundaydı. ASP.NET bu sorunun üstesinden gelir ve kullanıcısına çeşitli imkanlar sunar. Örneğin oturum durumuna ait ASP.NET’in sağladığı özellikleri beğenmiyorsanız silebilir ve kendi geliştirdiğiniz bir uygulamayı kullanabilirsiniz veya eksik buluyorsanız ASP.NET’in sunduğu imkanları inherit edip yeni özellikler katabilirsiniz. ASP.NET tamamen açıktır ve kullanıcı kendisine uygun olacak şekilde değiştirebilir.

Bileşen Kullanımı

ASP'de çoğu zaman kendi yazdığımız scriptler yetmiyordu ve uygulamalarımıza dışarıdan bileşenler eklememiz gerekiyordu. Bu hazır kodları kullanabilmemiz için de ekstra bir efor sarf etmemiz gerekiyordu. Öncelikle bu bileşenlerin sisteme kaydının yapılması gerekiyordu ve daha sonra bu bileşenlerin daha yeni sürümlerini kullanmak istediğimizde IIS'i durdurup güncellemeyi yapıp tekrar başlatmamız gerekiyordu. Bir Web sunucusu için bu hiç istenilmeyecek bir durumdur. Bir bileşenin konfigürasyonu da tam bir dertti. ASP.NET ile kayıt, versiyon ve konfigürasyon sorunlarının çoğu ortadan kalktı. ASP.NET , ASP gibi sadece COM objelerinin sunduğu arayüzü kullanmaz. ASP.NET sayfaları COM objeleri ile eskisine göre daha uyumlu çalışırlar. Bir COM dll'inin yeni bir versiyonunu yüklediğinizde xcopy ile ASP.NET, dosyaların güncellendiği hakkında uyarılır. Eski versiyona olan isteklerin cevaplanması bitene kadar yeni versiyon yüklenir ve bundan sonraki istekler yeni versiyona göre yapılır. Önceden çok karmaşık bir yapıda IIS Metabase'i içinde saklanan konfigürasyon bilgileri artık XML formatında saklandığı için insanlar tarafından okunup anlaşılması daha kolay bir hale geldi.

Her Kullanıcıda Çalışır

ASP'nin istemde bulunan her tür Web tarayıcısında çalıştığı söylenirdi fakat Internet Explorer, Netscape ve diğer tarayıcıların her birinin değişik versiyonları için bunun garantisini vermek zordu çünkü, ASP sayfaları HTML kodu üretir ve değişik tarayıcılar bu kodları farklı yorumlarlar. ASP.NET'de sunucu tarafında çalışan birçok bileşen vardır ve bunlar istemde bulunan Web tarayıcısını tanırlar ve onun anlayacağı şekilde çıktı üretirler. Hatta Mobil Internet Toolkit'i kullanırsanız istemde bulunan her tür cihaza göre çıktı üretebilirsiniz.

İstemci Durumunun İdaresi

ASP'nin istemci durumunun idaresi için sağladığı problemli ve yetersiz imkanları yerini çok daha fazla kullanıcıyı destekleyen, daha performanslı ve muli-threaded imkanlara bırakıyor. Örneğin bir kullanıcının durumunu SQL Server veritabanına kaydedebilmek gibi.

ASP İle Tam Uyum

Eskiden yazdığınız ASP kodlarını değiştirmeden kullanmak zorundaysanız hiç sorun değil, ASP.NET buna izin veriyor. ASP.NET, kullanıcılarına çok yumuşak bir geçiş imkanı tanıyor.

Kullanıcı Verimliliğindeki Artış

ASP.NET tamamen kullanıcının verimliliğini arttıracak şekilde dizayn edilmiştir. Dinamik Web sitelerini hazırlamak artık çok daha az zaman alıyor çünkü ASP.NET bir çok işi kendisi otomatik olarak yapıyor. Karmaşık bir çok işi sadece sürükleyip bırakarak yapabiliyoruz veya bir Web servisine referans tanımlayıp o Web servisinin sunduğu tüm imkanlardan yararlanabiliyoruz.

Konfigürasyon ve Mimari

ASP.NET ile konfigürasyon ve mimari çok daha güçlü bir hale getirilirken aynı zamanda çok da basitleştirilmiştir. Şimdi, bu konuda yapılan değişikliklerin neler olduğuna bakalım.

Konfigürasyon Dosyaları

Konfigürasyon dosyaları XML formatında olduğu için kullanıcılar tarafından okunup değiştirilebilmesi çok basit hale gelmiştir. Bu dosyaların yapıları ve içerikleri hakkında bilmemiz gereken bazı önemli noktalar vardır.

En üst seviyedeki konfigürasyon dosyası "machine.config" dosyasıdır ve <winnt>\Microsoft.NET\Framework\<version> klasörü içinde yer alır. Bir makinede çalışan tüm uygulamalar bu dosyadan etkilenirler. Diğer konfigürasyon dosyaları ise "web.config" dosyası adını taşırlar ve bu dosyalar her uygulamaya ait klasör içinde bulunurlar.

Konfigürasyon dosyalarının içerdikleri kayıtlar konfigürasyon dosyasının çeşidine göre değişir. Tüm makine için geçerli olan kayıtlar sadece "machine.config" dosyası içinde bulunur. Örneğin <processmodel>. Sadece uygulama seviyesinde geçerli olan kayıtlar ise hem <machine.config> hem de <web.config> dosyaları içinde bulunabilirler. Örneğin <sessionstate> gibi kayıtlar. Bu kayıtlar sadece

uygulama seviyesinde geçerli oldukları için daha alt seviyelere etkileri olmaz, sadece o uygulamaya ait klasör ve bu klasörün altındaki klasörler için geçerlidir. Bir uygulamaya ait klasörün alt klasörleri için de konfigürasyon dosyaları yaratılabilir. <security> güvenlik kaydı gibi kayıtlar her iki tür konfigürasyon dosyasında da bulunabilir çünkü bu tür bilgiler dosya ve klasör seviyesinde etkilidirler. Bir konfigürasyon dosyasının içeriğinin nasıl olabileceğine dair bir örnek aşağıda verilmiştir:

```
<configuration>
        <configsections>
            <!--config bölümüne ait alt elemanlar-->
        </configsections>

        <httpmodules>
            <!--http modülüne ait alt elemanlar-->
        </httpmodules>

        <httphandlers>
            <!--http tutucularına ait alt elemanlar-->
        </httphandlers>
        <sessionstate>
            <!—oturum durumuna ait alt elemanlar-->
        </sessionstate>
</configuration>
```

Aşağıda konfigürasyon dosyalarında kullanılabilecek ayarlar kısaca açıklanmıştır:

<appsettings> : Data Source Name gibi uygulama bilgilerini tanımlar.
<compilation> : ASP.NET için tüm derleme ayarlarını tutar.
<browsercaps> : Bir Web tarayıcısının yeteneklerini ayarlamada kullanılır.
<configuration> : Tüm konfigürasyon ayarlarını tutan konfigürasyon bölümüdür.
<customerrors> : Kullanıcının kendisine ait olan hata mesajlarını tanımlar.

<configsections> : Her config bölümü ile ilgili tutucuların bir listesi.
<globalization> : Bir uygulama için global anlamda geçerli olan ayarları düzenlemede kullanılır.
<location> : Bir klasör veya dosya gibi belirli bir yere ayarları gönderir.
<httpmodules> : HTTP modülleri ekleyip siler ve bu modüllerin içeriğini temizler.
<security> : Tüm güvenlik ayarlarını tanımlar.
<processmodel> : IIS için işlem modeli ayarlarını tutar.
<trace> : ASP.NET'teki trace servisini ayarlar.
<sessionstate> : Port numarası, çerez bilgileri (cookies) ve zaman sınırı gibi oturum durumunu ilgilendiren bilgileri ayarlamada kullanılır.
<webcontrols> : Kontrollerin kullanıcıdaki yerlerini tanımlar.
<webservices> : Web servisi ayarlarını kontrol eder.

Konfigürasyon dosyaları XML dokümanları oldukları için bu dosyaları kullanırken iyi düzenlenmiş XML dokümanları için geçerli olan kurallara uymak gerekir. [17]

Konfigürasyon dosyaları hiyerarşik bir yapıya sahiptir.

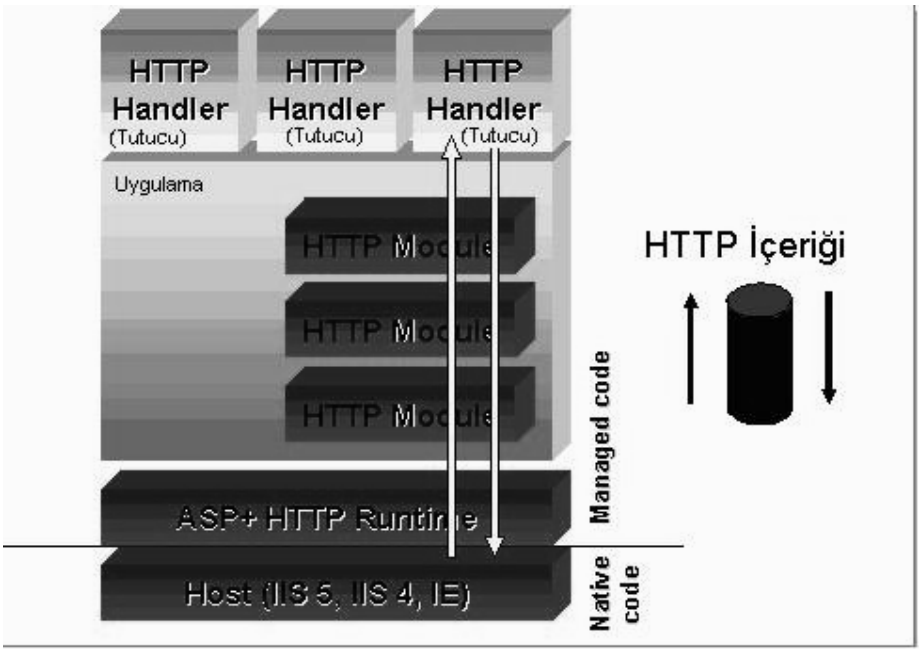
HTTP Runtime

HTTP Runtime (çalışma zamanı) ASP.NET'in üzerine kurulduğu, mesajlaşmayı sağlayan bir araç, bir motordur. Http Runtime, Http istemlerinin ve bu istemelere verilen yanıtların işlenmesini sağlayan alt yapıdır.

Http istemlerinden ve yanıtlarından bahsedebilmemiz için öncelikle bir bilgisayar üzerine kurulu bir hosta gereksinimimiz vardır. Bu genellikle Internet Information Server'dır. IIS, Http istemleri için 80 numaralı portu, Https istemleri için ise 443 numaralı portu dinler. Bir .aspx sayfası için gelen bir istek bir ISAPI dll'i olan XSPISAPI.dll tarafından engellenir. Bu DLL, ASP.NET çalışan işlemi olan XSPWP.exe'yi devreye sokar. Bu andan sonra yapılacak işleri bu uygulama düzenler. [17]

HTTP Runtime'ın Yapısı

Hostun Internet Information Server olması şart değildir, 80 numaralı portu dinleyen başka bir uygulama da olabilir. Örneğin üçüncü parti bir Web sunucusu veya kendi yazdığınız, aynı işi gören bir uygulama veya simüle edilmiş Web isteklerini yanıtlayan bir uygulama da olabilir.



HTTP Handler(tutucu)

Http Handler, mevcut sistemlerde kullanılmakta olan ISAPI'lere karşılık gelmektedir. Web'den gelen bir isteğin işlenmesindeki son duraktır. İstekte bulunulan sayfanın uzantısına göre ilgili tutucu o istek için ayrılır. Bir .aspx sayfasına gelen istekle ilgilenebilecek birçok tutucu olabilir fakat bunlardan sadece bir tanesi kullanılır. Tutucular, Http modüllerinin çalışmaları bittikten sonra çalışmaya başlarlar. Web.config dosyasının içerdiği konfigurasyon ayarlarına göre bir istekle ilgilenecek olan tutucu belirlenir. [17]

Http Module(modül)

Http modüllerinin mevcut sistemlerdeki karşılığı ISAPI filtreleridir. Bu modüller, her Web istemiyle birlikte çalıştırılması istenen kodları içeren sınıflardır. Örneğin, ön belleğe yükleme ve kullanıcı durumunun denetlenebilmesi gibi.

Web'den gelen her ASP.NET isteği için çalıştırılacak modüllerin listesini görebilmek için machine.config dosyasının <httpmodules> bölümüne bakabilirsiniz.

Global.asax (uygulama konfigurasyon dosyası)

Global.asax dosyaları bir uygulama kapsamında geçerli olan kodun yazılabileceği yerlerdir. Bu dosyalar uygulamanın kök dizini içinde yer almalıdırlar. Global.asax dosyaları birçok yararlı olay için tutucu (handler) içerebilirler. Bu uygulamaların bazıları aşağıda verilmiştir:

- Application_BeginRequest
- Application_AuthenticateRequest
- Application_AuthorizeRequest
- Application_ResolveRequestCache
- Application_AquireRequestState
- Application_PreRequestHandlerExecute
- Application_PostRequestHandlerExecute
- Application_ReleaseRequestState
- Application_UpdateRequestCache
- Application_EndRequest
- Session_OnStart
- Session_OnEnd

ASP.NET, bu olayların senkronizasyonuna ve kullanıcıların kendi tutucularını eklemelerine izin verir. Bu durum çok daha güçlü ve esnek uygulamaların geliştirilmesine olanak sağlar.[17]

3.3.3 Veri Madenciliğinde Sınıflama Ve Tahmin Yürütme

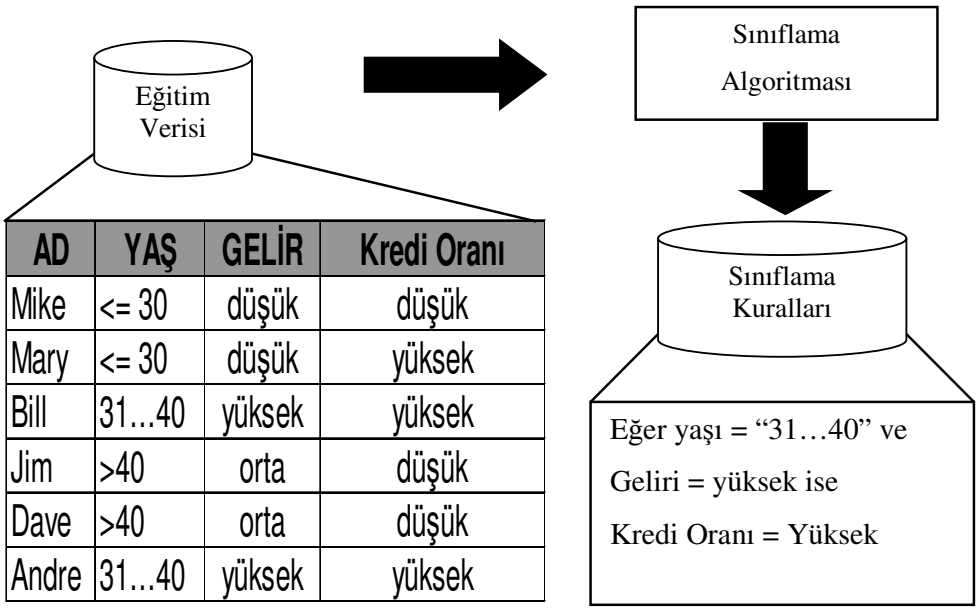
Veritabanları iş süreçlerinde karar analizlerinde çok gizli verilere sahip olabilirler. Sınıflandırma ve tahmin yürütme veri analiz yöntemlerinden biridir. Gelecek nesil veri analiz trendleri olarak söz edilmektedir. Fakat sınıflandırma ve tahmin yürütme birbirlerinden farklı olarak kendilerine farklı uygulamalarda yer

bulacaklardır. Örneğin sınıflandırma bir finansal uygulamada risk veya güvenlik üzerine uygulama alanı bulurken, tahmin yürütme müşterilerin gelir ve meslek dağılımlarına göre potansiyelleri belirleyen bir uygulama alanında kullanılabilir.
Bir çok sınıflandırma ve tahmin yürütme yöntemleri makine öğrenmesine, uzman sistemlere ve istatistiki yöntemlere dayanır.[18]

3.3.3.1 Sınıflandırma nedir? tahmin yürütme nedir?

Veri sınıflandırma iki adımlı bir işlemdir. İlk adım daha önceden belirlenip tanımlanmış olan veri kümeleri üzerinde bir model tanımlamaktır. Bu model tanımlaması kayıtların nitelikleri kullanılarak veri tabanı yapısından yola çıkılarak yapılır. Bir kayıt setin daha önce tanımlanmış olan bir sınıfa ait olduğunu varsayalım. Niteliklerden biri tarafından belirtilen sınıfa sınıf etiket niteliği adı verilir. Kayıt kümeleri eğitim veri kümesinin oluşturulmasında kullanılırlar. Bu eğitim verileri rasgele seçilir. Her bir sınıflandırma örneklendirme kümeleri geliştirilir. Bu adım denetlenmiş bilgi adımı olarak da (supervised learning) bilinir.

Tipik olarak, eğitim modeli karar ağaçları matematiksel formüller gibi sınıflandırma kuralları içinde gösterilir. Örnek olarak verilen müşteri kredi bilgileri bilgilerinde sınıflandırma kuralları müşteri kimliğine veya kredi oranlarına uygulanabilir. Kurallar gelecek zaman zarfında olası verileri kategorize edebilmek için kullanılabilir. [18]

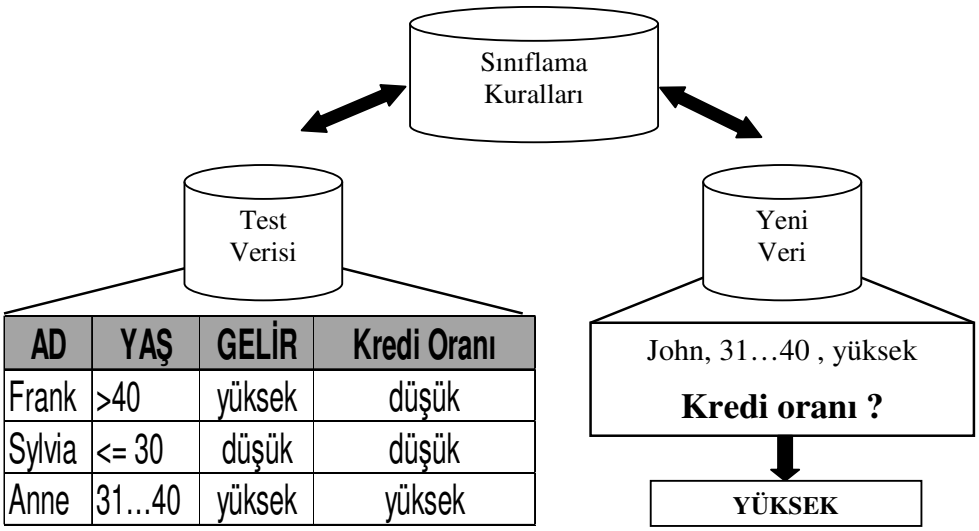


AD	YAŞ	GELİR	Kredi Oranı
Mike	<= 30	düşük	düşük
Mary	<= 30	düşük	yüksek
Bill	31...40	yüksek	yüksek
Jim	>40	orta	düşük
Dave	>40	orta	düşük
Andre	31...40	yüksek	yüksek

Şekil 3.12 Veri sınıflamada öğrenme

Şekil 3.12 de gösterildiği gibi ilk adım olarak veri kümesinin üzerinde bir sınıflandırma algoritması çalışır. Örnekte veri kümesinde ad soyad, yaş, gelir, kredi oranı gibi alanlar vardır. Sınıflandırma algoritmasında kullanılan sınıflandırma kuralları ise eğer müşterinin yaşı 30–40 aralığında ve geliri yüksek ise kredi oranı yüksektir gibi tanımlanmışlardır.

Bir banka önceki dönemlerde vermiş olduğu kredilere ilişkin gerekli tüm verilere sahip olabilir. Bu verilerde bağımsız değişkenler kredi alan müşterinin özellikleri, bağımlı değişken değeri ise kredinin geri ödenip ödenmediğidir. Bu verilere uygun olarak kurulan model, daha sonraki kredi taleplerinde müşteri özelliklerine göre verilecek olan kredinin geri ödenip ödenmeyeceğinin tahmininde kullanılmaktadır.[5]



Şekil 3.13 Veri sınıflamada sınıflama

Şekil 3.13 de ise test veriler üzerinde sınıflandırma kuralları uygulanarak yeni veriler elde edilmiştir. Eğer modelin doğruluğu kabul edilebilir ise bu model gelecek veri kümeleri için kullanılabilir. Örnek olarak şekil 3.13 deki kurallardan elde edilmiş sınıflandırma kuralları var olan müşterilerin yeni veya gelecekteki kredi oranlarını tahmin etmek için kullanılabilir. Hal böyleyken tahmin yürütme, kestirim nasıl oluyor da sınıflandırmadan ayrılıyor? Kestirim sınıflandırma için temel teşkil eder ve üzerinde çalışılan algoritmada değer biçme sürecinde devreye girer. Sınıflandırma ve gerileme iki temel kestirim problemidir. Sınıflandırma ve tahmin kendine kredilendirme uygulamalarında, tıbbi uygulamalarında ve performans analizi, satış analizi gibi uygulamalarda fazlaca yer bulmuştur.[18]

Örnek olarak;

YAŞ	GELİR	ÖĞRENCİ	Kredi Oranı	Bigisayar Aldı
<=30	yüksek	hayır	düşük	hayır
<=30	yüksek	hayır	yüksek	hayır
31…40	yüksek	hayır	düşük	evet
>40	orta	hayır	düşük	evet
>40	düşük	evet	düşük	evet
>40	düşük	evet	yüksek	hayır
31…40	düşük	evet	yüksek	evet
<=30	orta	hayır	düşük	hayır
<=30	düşük	evet	düşük	evet
>40	orta	evet	düşük	evet
<=30	orta	evet	yüksek	evet
31…40	orta	hayır	yüksek	evet
31…40	yüksek	evet	düşük	evet
>40	orta	hayır	yüksek	hayır

Tablo 3.1 Örnek test verileri

Allelectronics firmasına ait müşterilerin ev, iş adreslerini tutan veritabanına sahip olduğumuzu varsayalım. Bu adresler yeni bir ürün çıktığında, bir ürüne ait kampanya, indirim vs olduğunda müşterilere haber vermek için kullanılıyor olsun. Veritabanı tablosunda müşterinin adı soyadı, yaşı, geliri mesleği ve kredi oranı gibi bilgiler tutuluyor. Bu müşteriler Allelectronics firmasında bilgisayar satın almış olanlar ve satın almayanlar diye iki sınıfa ayrılabilir. Veritabanına yeni kaydedilen müşterilerin kısa zaman sonra gerçekleştirilecek bilgisayar satışlarındaki kampanya ile ilgili bilgilendirilmesini isteyebilirsiniz. Yeni olan tüm müşterilere mektup yazmak çok maliyetli olabilir. Bunun yerine sadece bilgisayar alabilecek sınıfına dahil olan müşterilere mektup yazmak daha uygundur. Bir sınıflandırma modeli bu yapıyı kurmak ve kullanmak amaçlı kullanılabilir. [18]

3.3.3.2 Sınıflandırma ve kestirim yayınlama

Bu bölümde veri sınıflandırma ve kestirim için yapılacak ön işlemler anlatılacaktır. Sınıflandırma ve değerlendirme için kriterlere de değinilecektir.

3.3.3.3 Verileri sınıflandırma ve kestirim için hazırlama

Sınıflandırma ve kestirim işlemlerinin doğruluğunun, uygulanabilirliğinin, etkinliğinin sağlanabilmesi için yapılması gereken adımlar aşağıda sıralanmıştır.

Veri Temizleme : Bu önişlem içinde veri içindeki tutarsızlıklar giderilir. Gürültüler temizlenir. Boş veya hatalı olan kayıtlar düzeltilir ya da silinir. Verinin tutarlı hale gelmesi sağlanır. Birçok sınıflandırma algoritması gürültülü ya da tutarsız veriler üzerinde çalışabilecek şekilde geliştirilmiştir. Ancak bu durumu performansı ve eğitim verilerini etkileyebilir.

İlişki Analizi : Veri içindeki bir çok alan sınıflandırma veya kestirim için ilgili bir alan olmayabilir. Örnek olarak müşterilerin meslek gruplarıyla alışveriş alışkanlıklarını sınıflandırmak istediğimizde adres alanları bizim işimize yaramayacaktır. Makine öğrenmesinde bu adım özellik seçimi (feature selection) olarak adlandırılır.

İdeal olan zamana bağlı olan ilişki analiz uygulamalarında eğitim seti alt kümelerine ayrılarak işlem yapılır. Yeni bir kayıt eklendiği zaman bunun öğrenilmesi daha önceki orijinal kayıtların öğrenilmesinden daha az zaman alacaktır. Bu gibi analiz metodları sınıflandırmanın daha etkili olmasını sağlar.

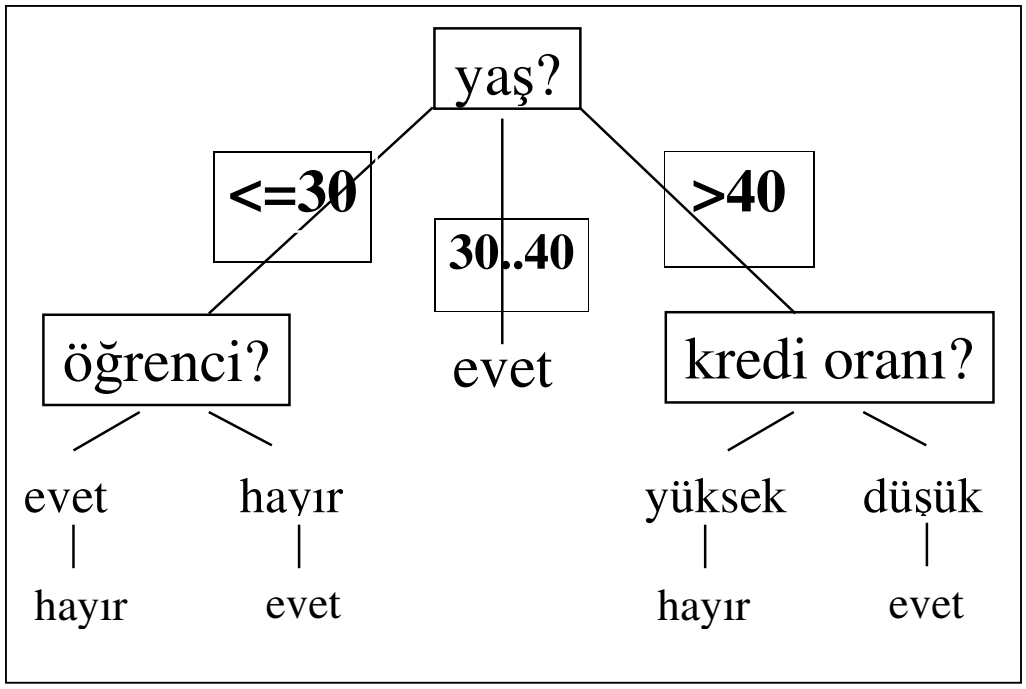
Veri Transferi : Kayıtlar kavram hiyerarşisinde daha üst seviyede bulunan kayıtlar ile genelleme yapılarak birleştirilebilir. Bu süreklilik gösteren kayıtların sınıflandırılmasında kullanılabilir. Örnek olarak sayısal değerler çok çeşitlilik arzedeceğinden bunlar düşük, orta yüksek gibi kavramlarla ifade edilerek genelleştirilebilirler. Benzer şekilde veri içerisinde bulunan cadde isimlerini illere göre bir üst hiyerarşide bulunan veri ile genellenebilirler.

Eğitim sırasında yapay sinir ağlarıda kullanılabilir. Bu sırada veriler normalize edilerek sınıflandırma işlemi gerçekleştirilmiş olabilir. [18]

3.3.3.4 Sınıflandırma yöntemlerinin karşılaştırılması

Sınıflandırma ve kestirim yöntemlerini karşılaştırmak için kullanabileceğimiz kriterler şunlardır.

Doğruluk Kestirimi : Verilerin sınıflantırılmasındaki yeteneği ifade eder.
Hız : Hesaplama performansı.
Sağlamlık : Verilen gürültülü veriler üzerindeki doğru sınıflandırma yeteneği.
Scalability : Ölçülebilirlik.
Açıklanabilirlik : Anlaşılırlık.



Şekil 3.14 "Bilgisayar alır" karar ağacı örneği

Şekil 3.14 de bir karar ağacı verilmiştir. Yaş grubuna göre müşterinin bilgisayar alma olasılığının varlığı karar ağacında belirtilmiştir. Buna göre yaş 31-40 arasında ise bilgisayar alma olasılığı vardır. Yaş 30 dan küçükse müşteri öğrenci ise olasılık varken öğrenci değil ise olasılık yoktur denilmiş. Yaş 40 den büyük ise bilgisayar alabilme olasılığı için kredi oranına bakılmış. Kredi oranı çok yüksekse olasılık var diye tanımlama yapılmıştır. [18]

3.3.3.4 Karar ağaçları ile sınıflandırma

Karar ağaçları akış şemasına benzer ağaç yapılarıdır. Her bir nitelik bir düğüm tarafından temsil edilir. Dallar ve yapraklar ağaç yapısının elemanlarıdır. En son yapı yaprak, en üst yapı kök ve bunların arasında kalan yapılar ise dal olarak adlandırılır. Şekil 3.14 de tipik bir karar ağacı verilmiştir. Bu yapıda müşterinin bilgisayar alabilme olasılığı işlenmiştir. Karar ağaçları sınıflandırma algoritmalarını uygulayabilmek için uygun bir yapıdır. Karar ağaçları birçok yerde kullanılmaktadır. Bunların başlıcaları sağlık, oyun ve iş sektörleridir. [18]

3.3.3.4.1 Karar ağaç yapıları

Karar ağaçlarında temel algoritma greedy algoritmasıdır. Bu algoritmada,

- Başlangıçta tek bir düğüm vardır ve eğitim örneğini temsil eder. Adım 1
- Eğer örneklerin hepsi aynı sınıfa aitse düğüm bir yaprak haline gelir. Adım 2
- Bu algoritma entropiye dayalı ölçüm kullanmakdatır. Hangi sınıfın üyesi olacağına karar verilir. Adım 3.
- Daha sonra bu nitelik test veya karar düğümü olur. Adım 4.
- Bilinen her bir test yaprağının değeri için bir dal oluşturulur. Ve bu değer için dallanma gerçekleştirilir. Adım 5.

Daha sonra algoritma rekürsif yapıda devam etmektedir.

3.3.3.4.2 Nitelik seçim ölçümü

En büyük entropi değerine sahip nitelik ağacın kökü durumuna getirilir. Entropi rastgeleliği, belirsizliği ve bekleyen bir durumun ortaya çıkma olasılığını verir. Bunun için istatiksel denklemler kullanılmaktadır. [18]

$$I(s_1, s_2, s_3, \text{K}\ , s_m) = -\sum_{i=1}^{m} p_i \log_2(p_i) \qquad \text{(i)}$$

$$E(A) = \sum_{j=1}^{v} \frac{Sij + \text{K} + Smj}{S} I(S_{1j}, \text{K}\ , S_{mj}) \qquad \text{(ii)}$$

$$Gain(A) = I(s_1, s_2, s_3, \text{K}, s_m) - E(A) \qquad \text{(iii)}$$

Karar ağacına bir örnek

Tablo 3.1 de Allelectronics firmasına ait müşteri bilgilerini tutan veri seti kümesi yapısı verilmiştir. Sınıflandırma sonucunda oluşan sınıfa "bilgisayar_alır" adı verilmiştir. Sınıflandırma iki tekil değere sahiptir. Bunlar evet-hayır parametreleridir. Evet kümesinde 9 örnek , hayır kümesinde 5 örnek vardır. Denklemde değişkenleri yerlerine koyarsak(i) ;

I(s1,s2) = I(9,5) = -(9/14)log2 9/14 - (5/14)log2(5/14) = 0.940 sonucunu elde ederiz.

Bir sonraki adımda her bir nitelik için entropi değerini hesaplamak gerekir. İlk olarak yaş niteliğinin hesaplamasıyla başlayalım. Yaş niteliğinde evet ve hayır olan değerlere bakalım. (ii) de verilen denklemi kullanarak değerimizi hesaplayalım.

Yaş = "<=30" : s11=2 , s21=3 , I(s11,s21) = 0,971

Yaş = "30…40": s12=4 , s22=0 , I(s12,s22) = 0

Yaş = ">40": s13=3 , s23=2 , I(s13,s23) = 0,971

E(yaş) = 5/14 I(s11,s21) + (4/14) I(s12,s22) + (5/14)I(s13,s23) = 0.694

(iii) Gain(yaş) = I(s1,s2)-E(yaş) = 0.246

Aynı denklemler ile diğer niteliklerin entropi değerlerini de hesaplayabiliriz.

Gain(gelir)= 0.029, Gain(öğrenci)= 0.151, Gain(kredi oranı)= 0.048 bulunur.

En büyük kazanç entropi değeri "yaş" niteliğinde olduğundan dolayı karar ağacı Şekil 3.14 teki yapıyı almıştır.

3.3.3.4.3 Ağaç budama

Yeni bir karar ağacı oluşturulduğu zaman veriler üzerindeki gürültüye bağlı olarak çok çeşitli ve çok sayıda yaprak ve dal oluşabilir. Budama işlemleri bu gereksiz ve ağaçta karmaşıklığa yol açan kayıtların silinmesi anlamına gelir.

Örnek olarak;

Eğer yaş = “<=30” ise öğrenci = “hayır” ise sonucu ulaşılabilir, öğrenci niteliği dışında “gelir” ve “kredi oranı” budamaya uğramıştır.

3.3.3.4.4 Karar ağaçlarından sınıflandırma kurallarına geçiş

Sınıflandırma kuralları karar ağaçlarından dışarı alınabilir mi? Alınabilirse nasıl yapılır? EĞER-İSE yapısı kullanılarak karar ağaçları içerisinde sınıflandırma kuralları sorgulanabilir. Ağacın kökünden bir yaprağa giden yol için bir eğer-ise kuralı tanımlanabilir.

Bir karar ağacından sınıflandırma kurallarını dışarıya almak. Şekil 3.14 de verilen yapıda ağacın kök hücresinden başlayarak yaprağa kadar gidilerek eğer – ise blokları elde edilir. Örnek olarak.

Eğer yaş = “<=30” ve öğrenci = hayır ise bilgisayar alır = “hayır”

Eğer yaş = “30…40” ise bilgisayar alır = “evet” birer kurallardır.

3.3.3.5 Bayesian sınıflandırması

“Bayesian sınıflandırıcıları nedir?” Bayesian sınıflandırıcıları istatiksel sınıflandırıcılardır. Özel bir sınıfa ait olarak verilen bir olasılık gibi, sınıf üyeliklerine ait olasılıklarını önceden söyleyebilirler.

Bayesian teoremi aşağıda verilen bayesian teoremini temel almaktadır. Sınıflandırma algoritmaları karşılaştırılarak yapılan çalışmalar naive(saf) Bayesian sınıflandırıcı olarak bilinen basit Bayesian sınıflandırıcıyı bulmuşlardır. Bu sınıflandırıcı karar ağaçları ve yapay sinir ağları sınıflandırıcılar ile karşılaştırılabilecek performansa gösteririler. Bunun yanında büyük veri tabanları üzerinde işlem yapan Bayesian sınıflandırıcılar yüksek hız ve doğruluk derecesine sahip olduklarını sergilediler.

Naive(saf) Bayesian, verilen bir sınıf üzerindeki bir özelliğe ait değerin etkisinin diğer özelliğe ait değerlerden bağımsız olduğunun farz eder. Bu kabul sınıf koşullu bağımsızlık olarak adlandırılır. Bu ise gerekli hesaplamaları basitleştirir,

anlaşılırlığı kolaylaştırır ve doğal olarak "saf" kelimesi ile ifade edilir. Bayesian belief ağları, Bayesian sınıflandırıcılardan farklı, özellik altkümeleri arasındaki ayrılıkları gösterilmesini sağlayan grafiksel modelledir. Bayesian belief ağlarıda sınıflandırma için kullanılabilmektedir. [18]

3.3.3.5.1 Bayes teorem

X sınıf üyeliği bilinmeyen veri örneği olsun. H ise bu veri örneği X in C sınıfına ait olduğunun öngören bir hipotez olsun. Bu sınıflandırma problemi için, biz P(H/X) olasılığını hesaplamak istiyoruz. Bu olasılık hipotez H' nın C sınıfına ait olarak farz edilerek verilen veriyi tutmasıdır.

P(H/X), H'ın X üzerindeki koşullandırmasına ait sonraki olasılıktır, *sonrasal olasılık.* Örneğin, veri örnek dünyamızın renk ve şekilleriyle tanımlanan meyveler olduğunu farz edelim. X' in kırmızı ve yuvarlak olduğunu ve H hipotezinin X' in bir elma olduğunu öngören bir hipotez olduğunu farz edelim. P(H/X), kırmızı ve yuvarlak olarak gördüğümüz X' in bir elma olduğunu güvenini(confidence) yansıtmaktadır. Karşıtlıkta, P(H), H' ye ait öncelikli olasılıktır, öncesel olasılık. Örneğimiz için, bu veri örneğinin görünüşüne aldırmadan verilen veri örneği herhangi birinin elmadır olasılığıdır. Sonrasal olasılık, P(H/X), X' den bağımsız olan öncesel olasılıktan, P(H), daha fazla bilgi üzerine kuruludur(örneğin arkaplan bilgisi).

Benzer şekilde, P(X/H) H' üzerinde koşul sağlayan sonraki olasılıktır. Bu ise, bize verilen X'in kırmızı ve yuvarlak olma olasılığı biliniyorsa bu elmadır diyebiliriz. P(X) X'in öncelik olasılığıdır. Bizim örneğimizi kullanarak, bu kırmızı ve yuvarlak meyvelere ait veri setinden veri örneği olasılığıdır.
"Bu olasılıklar nasıl kestirilir?" P(X), P(H), ve P(X/H) verilen veriden aşağıdaki şekilde kestirilebilir. Bayes teoremi sonraki olasılığı, P(H/X), P(H), P(X) ve P(X/H)'ı kullanarak hesap eden bir yol sağlayan kullanışlı bir teoremdir. [18]

Bayes teoremi $P(H/X) = \dfrac{P(X/H)P(H)}{P(X)}$, dir. (iv)

Şimdi, naive(saf) Bayesian sınıflandırıcı içinde Bayes teoreminin nasıl kullanıldığına bakabiliriz.

3.3.3.5.2 Naive(Saf) Bayesian sınıflandırması

Naive(saf) Bayesian sınıflandırıcı veya Bayesian sınıflandırıcı aşağıdaki gibi çalışır.

1. Her veri örneği n-boyutlu özellik vektörleri ile gösterilir, $X = (x_1, x_2, ..., x_n)$, her veri örneği n özelliklerden alınan örnek üzerindeki n ölçümler ile tarif edilir, sırasıyla, $A_1, A_2, ..., A_n$.

2. M sınıf olduğunu farz edin, $C_1, C_2, ..., C_m$. Verilen bilinmeyen bir veri örneğin, X(ör., bir sınıf üyeliğine sahip değil), sınıflandırıcı X'in en yüksek sonraki olasılığa sahip sınıfa ait olduğunu öngörecektir, X koşulunda. Sonuç olarak, naive(saf) Bayesian sınıflandırıcı bilinmeyen örnek X'i C_i sınıfına atar. Eğer ve sadece eğer ;
$P(C_i \mid X) > P(C_j \mid X)$ for $1 \le j \le m, j \ne i$.

Böylece $P(C_i \mid X)$'yi en yüksek büyüklüğe çıkarırız. C_i sınıfı $P(C_i \mid X)$'yi en yüksek büyüklüğe çıkarıldığı için *en büyük sonsal hipotez* diye adlandırılır. Bayes teoreminden (Eşitlik iv)

$$P(C_i / X) = \frac{P(X / C_i)P(C_i)}{P(X)} \qquad \text{(v)}$$

3. P(X) bütün sınıflar için sabit ise, sadece $P(X \mid C_i)P(C_i)$ en yüksek büyüklüğe çıkarılmasına ihtiyaç duyulur. Eğer sınıf öncelik olasılığı bilinmiyorsa, o zaman genel olarak sınıflar eşit kabul edilir, $P(C_1) = P(C_2) = ... = P(C_m)$, ve bu sebeple $P(X \mid C_i)$ ifadesini en yüksek büyüklüğe çıkarırız. Aksi taktirde, $P(X \mid C_i)P(C_i)$ ifadesi en yüksek büyüklüğe çıkarılır. Sınıf önceki olasılığı $P(C_i) = \frac{s_i}{s}$ olabilmektedir, burada s_i C_i sınıfına ait eğitilen örnek sayısı, ve s ise toplam eğitilen örnek sayısıdır.

4. Bir çok özelik barındıran veri setleri verilirse $P(X \mid C_i)$ aşırı derece hesap yükü gerektirir. $P(X \mid C_i)$ işleminde hesap yükünü azaltmak için, sınıf koşul bağımsızlığına ait saf varsayım uygulanır. Bu varsayım, özelliklere ait değerler bir diğerinden şartlı olarak bağımsızdır, örneğe ait verilen sınıf üyeliği, özellikler arasında bağımlılık ilişkisi yoktur. Bu,

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i). \qquad \text{(vi)}$$

Olasılıklar $P(x_1 \mid C_i), P(x_2 \mid C_i), ..., P(x_n \mid C_i)$ eğitim örneklerinde tahmin edilebilirler, burada

a) Eğer A_k kategorik ise, o zaman $P(x_k \mid C_i) = \frac{s_{ik}}{s}$, burada s_{ik} A_k için x_k değerine sahip olan C_i sınıfına ait eğitim seti sayısı ve s_i C_i'ye ait olan eğitim seti sayısıdır.
b) Eğer A_k devamlı-değer ise, özelliklerin tipik olarak Gauss dağılımına sahip oldukları varsayılır,

$$P(x_k \mid C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i}) = \frac{1}{\sqrt{2\pi\sigma_{C_i}}} e^{-\frac{(x_k \mu_{C_I})^2}{2\sigma_{C_i}^2}}, \qquad \text{(vii)}$$

burada $g(x_k, \mu_{C_i}, \sigma_{C_i})$ özellik A_k için Gauss yoğunluk fonksiyonudur, μ_{C_i} ve σ_{C_i} ortalama ve standart sapma iken, sırasıyla, C_i sınıfına ait eğitim örnekleri için A_k özellikleri için verilen değerler.

5. Bilinmeyen örnek X' i sınıflandırmak için, her C_i sınıfı $P(X \mid C_i)P(C_i)$ ifadesi hesaplanır. Örnek X'i C_i sınıfına atar eğer ve sadece eğer ;
$P(X \mid C_i)P(C_i) > P(X \mid C_j)P(C_j)$ for $1 \le j \le m, j \ne i.$

Diğer taraftan, $P(X \mid C_i)P(C_i)$ en yüksek değeri alması için C_i sınıfına atanır.

"Bayesian sınıflandırıcıları nasıl etkindirler?" Teoride, Bayesian sınıflandırıcılar diğer bütün sınıflandırıcılarla karşılaştırıldıklarında en düşük hata oranına sahiptirler. Bununla birlikte pratikte, kullanımı için yapılan varsayımlardaki hatadan dolayı olası bir durum değildir. Bununla birlikte, Karşılaştırma alanında yapılan bu sınıflandırıcı ile ilgili birçok deneysel çalışmalar bu sınıflandırıcının karar ağaçları ve yapay sinir ağları ile birçok alanda yarışabileceğini bulmuştur. [18]

Bayesian sınıflandırıcı Bayes teoremini kullanmayan diğer sınıflandırıcıları teorik olarak doğrulamak için de kullanılmaktadır. Örneğin, gerçek varsayım altında, birçok yapay sinir ağı ve kavis-uydurma algoritması çıkış olarak, naive(saf) Bayesian sınıflandırıcını yaptığı gibi, *en yüksek büyüklük sonrasal* hipotezini vermektedir.

Örnek olarak Saf (naive) Bayesian sınıflandırması kullanarak bir sınıf üyeliğinin bulunması: Karar ağaçları girişi için gösterilen örnekteki eğitim verisi içindeki bilinmeyen örneklerin sınıf üyeliklerinin saf (naive) Bayesian sınıflandırma

kullanılarak tahmin etmek istiyoruz. Eğitim verisi Tablo 3.1 'deki veridir. Veri örnekleri yaş*, gelir, öğrenci* ve *kredi_kullanımı* özellikleri tarafından tanımlanır. Sınıf üyelik özellikleri, *bilgisayar_alma,* ([evet, hayır] olarak adlandırılan) iki ayrı değere sahiptir. C_1, *bilgisayar_alama="evet"* değerine ve C_2, *bilgisayar_alma="hayır"* değerine eşit olsun. X = (yaş = "<=30", gelir = "ortalama", öğrenci = "evet", kredi_kullanımı = "yüksek") sınıflandırmak istediğimiz bilinmeyen örnektir. [18]

$P(X \mid C_i)P(C_i)$ değerini maksimize etmeye ihtiyacımız var, i = 1,2.P(C_i) için, her sınıf için önsel olasılık eğitim örneklerinden hesaplanabilir:

P(*bilgisayar_satınlama = "evet"*) = 9/14 =0.643
P(*bilgisayar_satınlama = "hayır"*) = 5/14 =0.357

$P(X \mid C_i)$ değerinin hesap etmek için, i =1,2 için, aşağıdaki koşullu olasılıkları hesaplayacağız:

P(yaş = "<30" | *bilgisayar_alma = "evet"*) = 7/9 = 0.777
P(yaş = "<30" | *bilgisayar_alma = "hayır"*) = 2/5 = 0.400
P(gelir = "ortalama" | *bilgisayar_alma = "evet"*) = 4/9 = 0.444
P(gelir = "ortalama" | *bilgisayar_alma = "hayır"*) = 2/5 = 0.400
P(öğrenci = "evet" | *bilgisayar_alma = "evet"*) = 6/9 = 0.667
P(öğrenci = "evet" | *bilgisayar_alma = "hayır"*) = 1/5 = 0.200
P(kredi_kullanımı = "yüksek"| *bilgisayar_alma = "evet"*) = 6/9 = 0.667
P(kredi_kullanımı = "yüksek"| *bilgisayar_alma = "hayır"*) = 2/5 = 0.400

Yukarıdaki olasılıklar kullanılarak;

P(*bilgisayar_satınlama = "evet"*) = 0.777*0.444*0.667*0.667 = 0,153
P(*bilgisayar_satınlama = "hayır"*) = 0.400*0.400*0.200*0.400 = 0,0128
P(*bilgisayar_satınlama = "evet"*) P(*bilgisayar_satınlama = "evet"*) = 0.153*0.643 = 0.0983

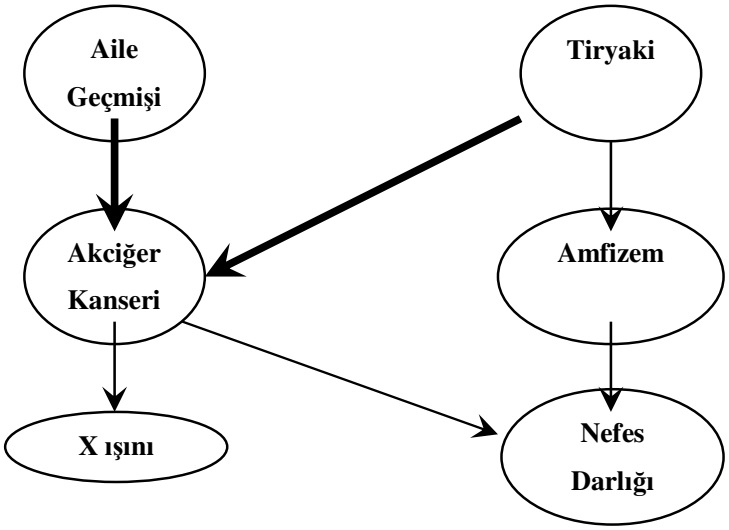
P(*bilgisayar_satınlama=“hayır”*)P(*bilgisayar_satınlama= “hayır”*)=0.0128*0.357= 0.004

Bu sebeple, saf(naive) Bayesian sınıflandırıcı *bilgisayar_alma = “yes”* kuralını örnek X’e göre tahmin eder.

3.3.3.5.3 Bayesian Belief ağları

Saf Bayesian sınıflandırıcı sınıf koşullu bağımsızlığı hakkında varsayım yapar, yani, örneklere ait verilen sınıf üyelikleri, özelliklere ait değerler bir diğerine koşullu bağımsızdır. Bu varsayım hesaplamaları basitleştirir. Varsayım doğruysa, Bayesian sınıflandırıcı diğer bütün sınıflandırıcılarla karşılaştırıldığında en doğru sonucu verendir. Bununla beraber, pratikte değişkenler arasında bağımlılıklar olabilir. Bayesian belief ağları ortak koşullu olasılık dağılımları belirtir ve değişkenlere ait alt kümeler arasındaki sınıf koşullu bağımsızlıkları tanımlamak için olanak sağlarlar. Ayrıca öğrenme sırasında nedensel ilişkilere ait grafiksel model sağlarlar. Bu ağlar belief ağları, Bayesian ağları, ve olasılıksal ağlar olarak bilinirler. Kısaltma için bu ağlara belief ağlar olarak da adlandırılırlar.

Bir belief ağı iki bileşen ile tanımlanır. İlki *yönlendirilmiş cevrimsiz çizelge*, burada her düğüm bir rasgele değişken ile gösterilir ve her yay (arc) bir olasılıksal bağımlılıkla gösterilir. Eğer bir yay düğüm Y’den düğüm Z’ye doğru çizilirse, o zaman Y Z’nin ebeveyni veya şu anki selefidir ve Z ise Y nin alt üyesidir. Her değişken çizelgedeki alt üyeleri olmayanlara koşullu bağımsızdır. Bu değişkenler ayrık veya sürekli değerler olabilirler. Sunulan veri içindeki gerçek özelliklere karşılık gelebilirler veya ilişkilerden gelen “gizli değişkenler” olabilirler (örneğin tıbbi belirti tıbbi veri ile ilgilidir). [18]



Şekil 3.15 Belief ağı

Şekil 3.15 basit bir belief ağını gösterir. Yaylar nedensel bilgiye ait bir gösterime izin verir. Örneğin, kişinin akciğer kanseri olmasını ailesinin geçmişindeki akciğer kanseri vakaları ve de kişinin sigara içip içmediği etkiler. Ayrıca, ebeveyni, *aile geçmişi* ve *sigara tiryakisi* tarafından verilen, yaylar amfizeme ait koşullu bağımsızlık olan *Akciğer kanseri* değişkenini de gösterir. Bunun anlamı başta *Aile Geçmişi* ve *sigara tiryakiliği* bilinmiyor ve Amfizem değişkeni *Akciğer Kanseri* ile ilgili ek bilgileri desteklemez.

	aile,sigara	Aile,~sigara	~aile,sigara	~aile,~sigara
A.Kanseri	0,8	0,5	0,7	0,1
~A.Kanseri	0,2	0,5	0,3	0,9

Tablo 3.2 Akciğer kanseri koşullu olasılılık tablosu

Bir belief ağında tanımlı ikinci bileşen her değişken için bir *koşul olasılık tablosu(CPT)* içerir. CPT bir Z değişkeni için koşul dağılımı P(Z|Ebeveyninler(Z)) tanımlar, burada Ebeveynler(Z) Z' nin ebeveynleridir. Tablo 3.2 *Akciğer Kanseri* için bir CPT'yi göstermektedir. Akciğer kanserine ait her değer için koşullu olasılık ebeveynlerin değerlerine ait verilen her olası koşuldur. Örneğin, üstte en soldaki ve altta en sağdaki girdilerden, sırasıyla,

P(AkcigerKanseri = “evet”| AileGecmişi = “evet”, SigaraTiryakisi = “evet”) = 0.8
P(AkcigerKanseri = “hayır”| AileGecmişi = “hayır”, SigaraTiryakisi = “hayır”) = 0.9

Herhangi bir satıra ait bileşik olasılık $(z_1,...,z_n)$ aşağıdaki gibi hesaplanan değişkenlere veya özelliklere $Z_1,...Z_n$ karşılık gelir.

$$P(z_1,...,z_n) = \prod_{i=1}^{n} P(z_i \mid Ebebeyn(Z_i)), \qquad \text{(viii)}$$

Burada $P(z_i \mid Ebebeyn(Z_i))$ değeri Z_i için CPT içindeki girdilere karşılık gelir. Sınıf üyelik özellikleri sunulurken, ağ içindeki bir düğüm “çıkış” düğümü olarak seçilebilir. Birden çok çıkış düğümü olabilir. Öğrenim için çıkarım algoritmaları ağ üzerinde kabul edilebilir. Sınıflandırma işlemi, tek sınıf üyelik dönüşümünden ziyade, sınıf üyelik özellikleri için bir olasılık dağılımı döndürürler, yani, her sınıfa ait olasılık tahminidir. [18]

7.4.4 Bayesian Belief Ağların Eğitimi

“*Bir Bayesian ağı nasıl eğitilir?*” Bir Bayesian ağının eğitimi veya eğitiminde, senaryolar olasıdır. Veriden ağın yapısını çıkarmak mümkün olabilmektedir. Ağ değişkenleri bütün veya birkaç eğitim seti içinde gizli veya gözlenebilir olabilmektedir. Saklı veri *kayıp değerler* veya *eksik verilere* işaret edebilir. Eğer ağ yapısı biliniyorsa ve değişkenler görünürse o zaman eğitilen ağ oldukça doğru sonuç verir. CPT girdilerinin hesaplanmasından meydana gelmektedir, saf Bayesian sınıflandırmadaki gereken olasılıkların hesaplanmasında olduğu gibidir. Ağ yapısı verildiğinde ve bazı değişkenler saklı, o zaman gradyan descent metodu belief ağının eğitimi için kullanılabilir. Amaç CPT girdileri için değerlerin öğrenilmesidir. s eğitim örneklerine ait bir S seti olsun, $X_1, X_2,..., X_s$. w_{ijk} $U_i = u_{ik}$ ebeveynlerine sahip $Y_i = y_{ij}$ değişkenleri için bir CPT girdisi olsun. Örneğin, eğer w_{ijk} tablo 3.2’deki üst en soldaki CPT girdisi ise o zaman Y_i

akciğerkanseridir; değeri y_{ij},"evet"; U_i Y_i'ait ebeveyn düğümlerini listeler, yani, *{AileGecmişi, SigaraTiryakisi}*; ve u_{ik} ebeveyn düğümlerin değerlerinin listeler, yani *{"yes","yes"}*. w_{ijk} ağırlık olarak görülebilir, yapay sinir ağlarındaki saklı katmana ait ağırlıklara benzetilebilir. Ağırlık seti müşterek olarak w ile gösterilir. Ağırlıklar rasgele olasılıksal değerler ile başlatılır. Gradyan descent stratejisi hırslı tepe-tırmanışını gerçekleştirir. Her iterasyon için, ağırlıklara güncelleştirilir ve nihayetinde optimum çözüme yakınsayacaktır. [18]

Bu metot en iyi veri modeli olan w_{ijk} değerleri için, w'ye ait her olası ayarlama olan varsayım üzerinde arama yapar. Bunun amacı $P_w(S) = \prod_{d=1}^{s} P_w(X_d)$ değerinin maksimum yapmaktır. Bu problemi basitleştiren $\ln P_w(S)$ gradyanı yolu izlenerek yapılabilir. Verilen ağ yapısı ve w_{ijk} başlangıç değerleri ile algoritma aşağıdaki şekilde yürütülür:

1. Gradyan hesaplaması: Her i,j,k için

$$\frac{\partial \ln P_w(S)}{\partial w_{ijk}} = \sum_{d=1}^{s} \frac{P(Y_i = y_{ij}, U_i = u_{ik} \mid X_d)}{w_{ijk}} \qquad \text{(ix)}$$

Eşitliğin (ix) sağ tarafındaki olasılık S içindeki her X_d eğitim seti için hesaplanacaktır. Kısalık için, bu olasılık basitçe p ile gösterilir. Bazı X_d için $Y_i ve U_i$ ile gösterilen değişkenler saklı olduğunda, Bayesian ağı çıkarsaması için basit standart algoritmalara ait gözlenebilen değişkenler kullanımından mukabil olasılık p hesaplanabilir.

2. Gradyan yönünde küçük adımlar atmak: Ağırlıklar aşağıdaki şekilde güncellenir,

$$w_{ijk} <-- w_{ijk} + (l)\frac{\partial \ln P_w(S)}{\partial w_{ijk}}, \qquad \text{(x)}$$

burada l adım boyu olarak sunulan öğrenme oranıdır, ve $\frac{\partial \ln P_w(S)}{\partial w_{ijk}}$ (x) eşitliğinden hesaplanır. Öğrenme oranı küçük sabitti.

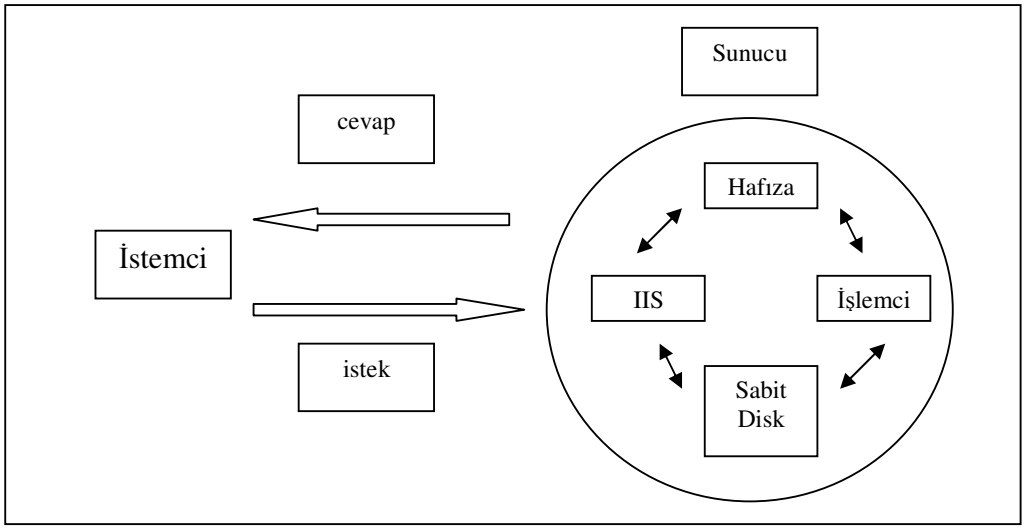
3. Ağırlıkların yeniden normalleştirilmesi: Çünkü ağırlıklar w_{ijk} olasılık değişkenleridir, 0,0 ile 1,0 arasında olmalıdırlar ve $\sum_j w_{ijk}$ bütün i,j,k için 1'e eşit olmalıdır. Bu kriter eşitlik (x) tarafından güncellendikten sonra ağırlıklar tekrar normalleştirilerek sağlanır.

Görülen özelliklerden elde edilen veri eğitiminden ağ yapısının öğretimim için birçok algoritma vardır. Problem ayrık optimizasyonlardan biridir. [18]

4. WEB PERFORMANSI ANALİZİNDE KULLANILAN YÖNTEM VE ARAÇLAR

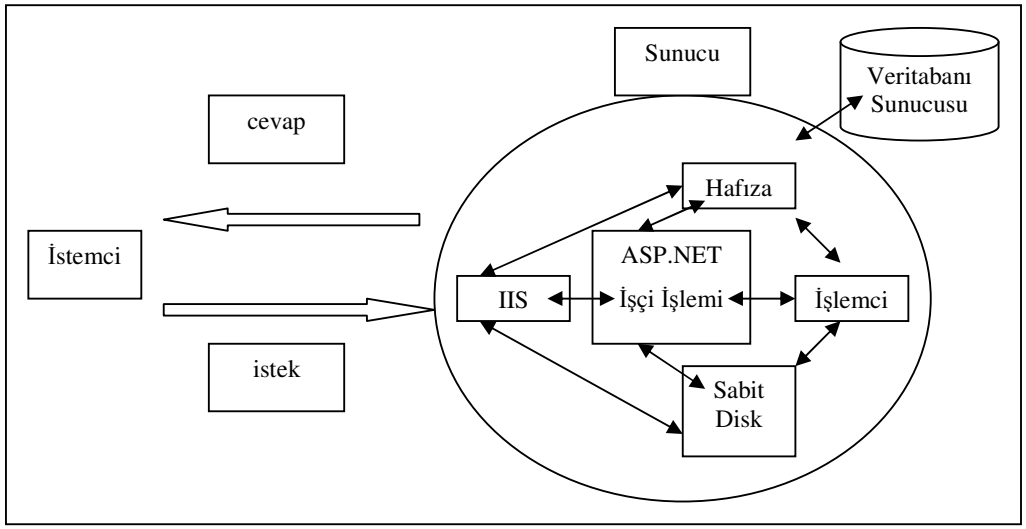
Web tabanlı sistemlerde kullanıcıların beklentileri yapmak istedikleri işlemleri hızlı ve sorunsuz yapmak istemeleridir, bu sayede aynı işi yapan diğer web tabanlı uygulamalarla mukayese edildiğinde ziyaret ettikleri web sayfasının performansı daha iyi ortaya çıkmakta ve sayfayı tekrar ziyaret etmek istemektedirler. Uygulama geliştirici tarafından ise konu değerlendirildiğinde yaptığı farklı yöntem programlama yapılarıyla dinamik, statik öğelerin kullanım oranı ve gerekliliği veritabanı bağlantısının gerekliliğinin tespit edilmesidir. Bu sayede çok çeşitli kodlama ve script yapıları kullanarak uygulamada oluşabilecek kodlama hatası, gereksiz kodların ayıklanması ve oluşabilecek darboğazlar önceden giderilmiş olur. Sistem yöneticisi tarafından ise sunucu sistem kaynakları en iyi şekilde kullanılması sağlamak ve uygulama yoğun yük altındayken bile sistemde meydana gelebilecek donanım darboğazları önlemeye yönelik olmaktadır.

Web sayfası, uygulamasının performansı farklı yöntemlerle ölçülebilmektedir. Burada kullanılan yöntemde sunucunun ortalama işlemci kullanım yüzdesi, sunucunun uygulamanın çalıştırılması sonucu hafızadaki kullanım artış yüzdesi değerlendirilmiş ve ölçülmüş. Web uygulaması tarafında ise web sayfasının boyutu, aynı anda maksimum hizmet edebileceği saniyedeki talep sayısı, dinamik öğe, statik öğe ve veritabanı bağlantısının olup olmaması gibi değerler ölçülerek, web sunucunun hizmet edebileceği web uygulamalarının profili oluşturmaya çalışılmıştır. Veri madenciliği konusu Saf Bayes yöntemiyle de bu profilin karşılaşmadığı web uygulamalarının performans ölçümlerini sınıflamaya yönelik tahmin sonuçları elde edilmiştir.



Şekil 4.1 Statik web uygulama çalışması

Şekil 4.1 de görüldüğü gibi statik web programcılığıyla geliştirilen web sayfalarında tipik olarak HTML ve istemci taraflı Javascript kodları kullanılmıştır. Javascript kullanmamızın nedeni birçok internet görüntüleyicisi(browser) ile düzgün çalışması ve istemci taraflı uygulamalarda çok yoğun olarak kullanılmasıdır.



Şekil 4.2 Dinamik web uygulama çalışması

Şekil 4.2 de tipik ASP.NET kullanılarak yapılan dinamik web programcılığının çalışması gösterilmektedir. Uygulamalarda ASP.NET' in tercih edilmesinin birçok nedeni vardır bunlardan başlıcalar;

- Web uygulamaları geliştirmede çok fazla popüler olması ve kullanılması.

- Web uygulamaları geliştirirken dilden bağımsız olması ve her platform ve cihaz için çıktı üretebilmesi.

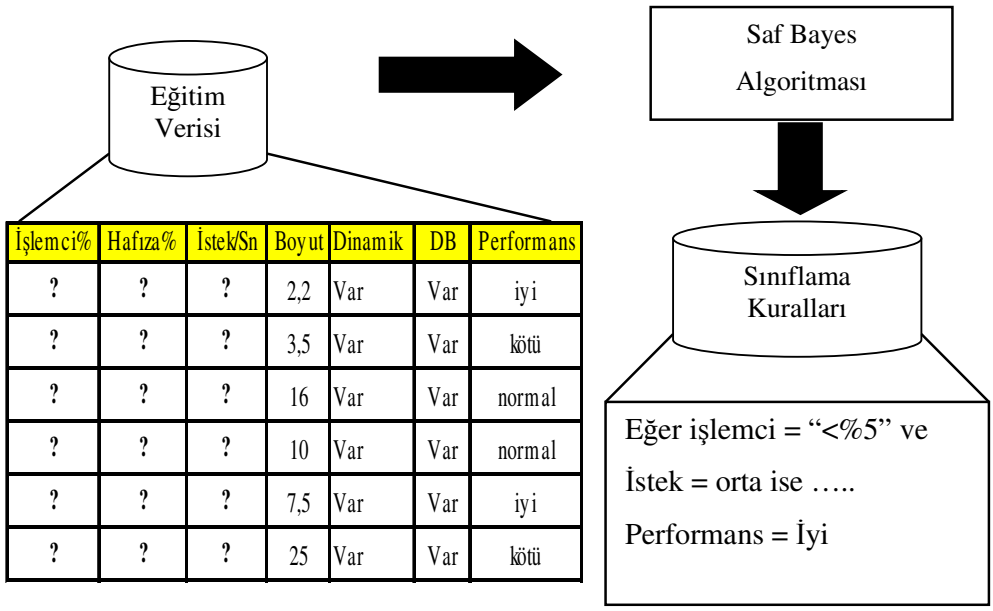
- Geliştirirken daha az kod gerektiriyor olması. Satır sayısında %40 ile %70 arasında azalma olur. Önceden HTML ve script kullanarak yapabildiğimiz işlerin birçoğunu artık sadece bir sunucu kontrolü koyarak yapabiliriz.

- İçerik ve kod birbirinden tamamen ayrılır ve farklı dosyalardadır

- Web uygulamalarının veritabanları ile kolay bağlantı kurmasına izin verir.

4.2 Eğitim Veri Kümesinin Oluşturulması

Saf bayes uygulamasının doğru sonuçları tahmin etmesi için ilk başta sınıflama kurallarının oluşturulması ve modelin eğitilmesi gerekmektedir. Sınıflama kuralları ise tahmin modelinin doğru sonuçlar üretmesi için gerekli bir adımdır. Elimizde var olan bilgiler şekil 4.3 teki tablodaki gibidir. Tablodaki soru işaretli yerlerin bulunması için her bir uygulamanın performans testinin yapılması gerekmektedir.



İşlemci%	Hafıza%	İstek/Sn	Boyut	Dinamik	DB	Performans
?	?	?	2,2	Var	Var	iyi
?	?	?	3,5	Var	Var	kötü
?	?	?	16	Var	Var	normal
?	?	?	10	Var	Var	normal
?	?	?	7,5	Var	Var	iyi
?	?	?	25	Var	Var	kötü

Şekil 4.3 Bayes uygulaması eğitim modeli

Uygulamamızda saf bayes algoritmasının tercih edilme nedenleri olarak;

- Bu sınıflandırıcı karar ağaçları ve yapay sinir ağları sınıflandırıcılar ile kıyaslanabilecek güçlü performans göstermesi.

- Büyük veri tabanları üzerinde işlem yapan bayes sınıflandırıcılar yüksek hız ve doğruluk deresine sahip olduklarından.

- Saf bayes, verilen bir sınıf üzerindeki bir özelliğe ait değerin etkisinin diğer özelliğe ait değerlerden bağımsız olduğunun farz eder. Bu kabul sınıf koşullu bağımsızlık olarak adlandırılır. Bu ise gerekli hesaplamaları basitleştirir, anlaşılırlığı kolaylaştırır ve doğal olarak "saf" kelimesi ile ifade edilir.
- Teoride, Bayes sınıflandırıcılar diğer bütün sınıflandırıcılarla karşılaştırıldıklarında en düşük hata oranına sahiptirler. Bununla birlikte pratikte, kullanımı için yapılan varsayımlardaki hatadan dolayı olası bir durum değildir. Bununla birlikte, Karşılaştırma alanında yapılan bu sınıflandırıcı ile ilgili birçok deneysel çalışmalar bu sınıflandırıcının karar

ağaçları ve yapay sinir ağları ile birçok alanda yarışabileceğini bulmuştur.[18]

Analiz modelinde eğitim verisi oluşturmak için dinamik, statik öğeler içeren farklı boyutlardaki 6 web sayfası oluşturuldu. Bu web sayfaları aşağıdaki Tablo 4.1' de özetlenmiştir.

Uygulama Adı	Uygulama Yöntemi	Boyutu (KB)
Sayfa1.html	Bu uygulama sadece statik öğeler içerir. Bu sayfa kendisine ulaşan öğrencinin açılır menüdeki seçimi istemci tarafındaki script koduyla kontrol eder. Öğrencinin bölümüne ait bölüm ders sayfaları görüntülenir.	2.2
Sayfa2.aspx	Bu uygulama sadece dinamik öğeler içerir. Açılan menü seçimini sunucu tarafında kontrol edilerek öğrencinin bölümüne ait bölüm ders sayfaları gösterilir.	3.5
Sayfa3.aspx	Bu uygulama dinamik ögeler ve veritabanı bağlantısı içerir. Girilen kullanıcı adına karşılık gelen bölüm dersleri veritabanındaki 1000 kişilik kullanıcı tablosundan sorgulanarak alınır. Eğer doğruysa bölümüne ait ders sayfaları gösterilir.	16
Sayfa4.html	Bu uygulama sadece statik öğeler içerir. Bu sayfa kendisine ulaşan kullanıcının seçimine göre bir alışveriş sitesinde kategorilerdeki ürünleri listeleyerek görüntülenmesini sağlar.	10
Sayfa5.aspx	Bu uygulama sadece dinamik öğeler içerir. Kullanıcının seçimi sunucu tarafında değerlendirilir ve alışveriş sitesindeki istenen kategoriye ait ürünler listelenir.	7.5
Sayfa6.aspx	Alışveriş sitesinden istenen kategorideki ürünler 20 kategori ve 500 ürün bulunan veritabanından sorgulanarak listelenir.	25

Tablo 4.1 Saf Bayes Eğitimi Sayfaları

Test eğitim sayfalarının sunucuda sistem kaynaklarını nasıl ve ne derecede kullandığını ölçmek için Microsoft firmasının geliştirdiği Web Application Stress Tool (WAS) ve Windows işletim sisteminde dahili bulunan Performance Monitor (perfmon.exe) performans izleme aracı kullanıldı.[6] Was programı web uygulamasını belirlenen bir zaman aralığı içerisinde istenen kullanıcı sayısı kadar sunucudan talep, gerçeğe yakın web senaryosu oluşturulmasına imkan verir. Performans izleme aracı ise sunucu kaynaklarının durumlarını izlemek için sayaçlar barındırır. Saf bayes tahmini için sunucunun her bir web uygulama sayfasının Was ile testi sırasında Perfmon ile ortalama işlemci % kullanım oranı, hafıza kullanım miktarındaki % değişim miktarı ve istek sayıları sayaçları kaydedilmektedir.
Sonuçları değerlendirmek için Perfmon' da kullanılan sayaçlar şunlardır:

- Processor.% Precessor Time = İşlemcinin boş olmayan bir iş parçacığını yürütmek için harcadığı süre yüzdesidir.
- Memory.% Committed Bytes In Use = Bayt sayacının Bellek\\Kaydetme Sınırı sayacına oranıdır. Kullanılan hafıza yüzdesi.
- Web Service.Total Get Requests = Hizmetin başlamasından itibaren ALMA yöntemini kullanan HTTP isteği sayısıdır.
- Web Service.Current Connections = Web sunucusuna bağlı kullanıcı anlık sayısını gösterir.

Company	Product	Virtual User	Price (USD)
Mercury Interactive	Loadrunner	> 50	> 40.000
RSW Software	E-Load	> 100	> 25.000
Segue Software	Silkperformer	> 100	> 25.000
Microsoft	Web Stress Tool	Unlimited	0

Tablo 4.2 Web Stress Tool'un Tercih Nedeni

Performans testi değerlerini ölçmek için kullanılan istemci özellikleri Şekil 4.4 te verilmiştir.

Donanım	Yazılım
Pentium 4 3.2Ghz işlemci 1 GB ram 100 Mbps Ethernet LAN	Windows xp pro işletim sistemi Web Application Stres Tool 75 Kullanıcı ve 56 Kbps ayarlı

Şekil 4.4 İstemci Özellikleri

Was'ın ayarlanmasında 75 kullanıcının seçilmesinin nedeni *sayfa3.aspx* web uygulamasının en fazla 75 kullanıcıya hizmet verebilmesinden kaynaklanmıştır. Eğer 75 ten fazla sanal kullanıcı tanımlanırsa sunucunun donanım sistemleri cevap verememekte ve sunucudan sayfa bulunamadı ya da iç sunucu hatası mesajı alınmaktadır. Bunun yanında test için 3 dakikanın seçilmesi ise sunucuya talepleri cevaplaması için yeterli süreyi tanımak ve sayaç değerlerinin doğru sonuçları vermesi sağlamaktır. Was ile testlerde ilk 0.5 dakika sunucuya ısınma(warm up)

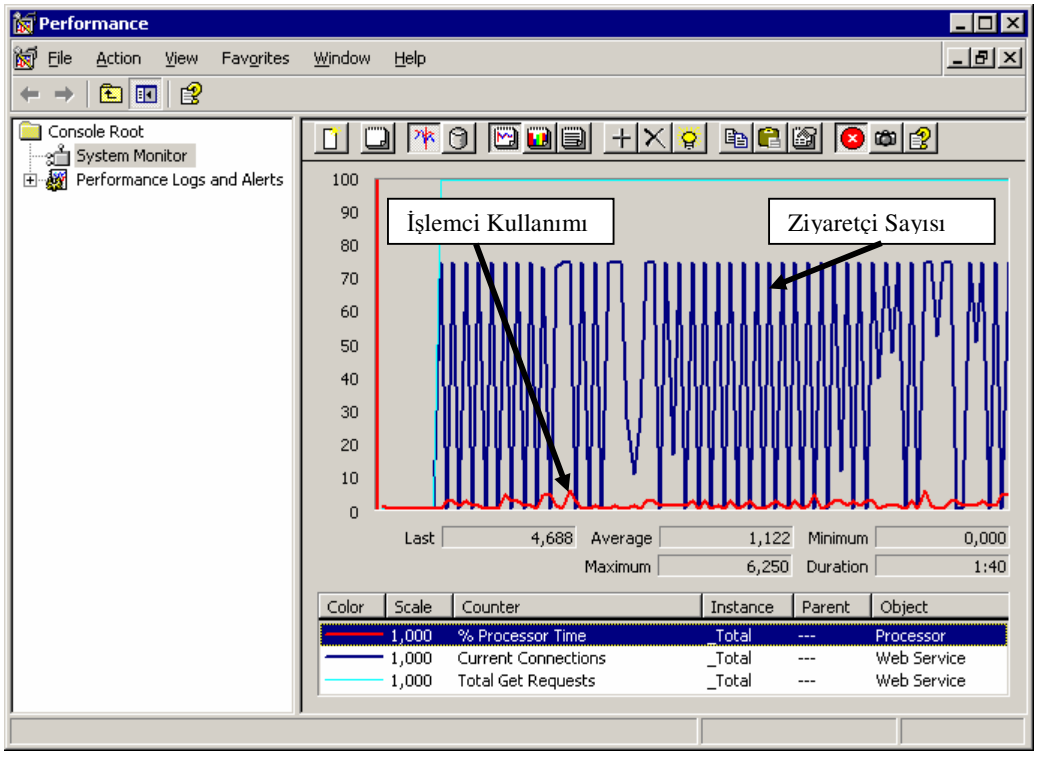
süresi son 0.5 dakika soğuma(cool down) süresi tanımlanmıştır. Böylece testlerden elde edilen grafiklerde ilk sayaç grafikleri sunucuda yük yok iken görülmesi sağlanmıştır.

Donanım	Yazılım
Pentium 4 3.2Ghz işlemci 1 GB ram 100 Mbps Ethernet LAN	Windows 2003 server işletim sistemi Internet Information Server 6.0 Framework 1.1 Performance Monitor

Şekil 4.5 Sunucu Özellikleri

4.4 Test 4.1

Tablo 4.1 de verilen sayfa1.html sadece statik öğelerden oluşan web uygulaması was programı ile sunucudan talep edilmesine bağlı olarak sunucu kaynakları test edilmiştir. Toplam 3 dakika ve 56 kbps sahip 75 kullanıcının aynı anda bağlanmasıyla oluşan test sonucunda Performance Monitor'den elde edilen grafik verileri aşağıdaki gibidir.



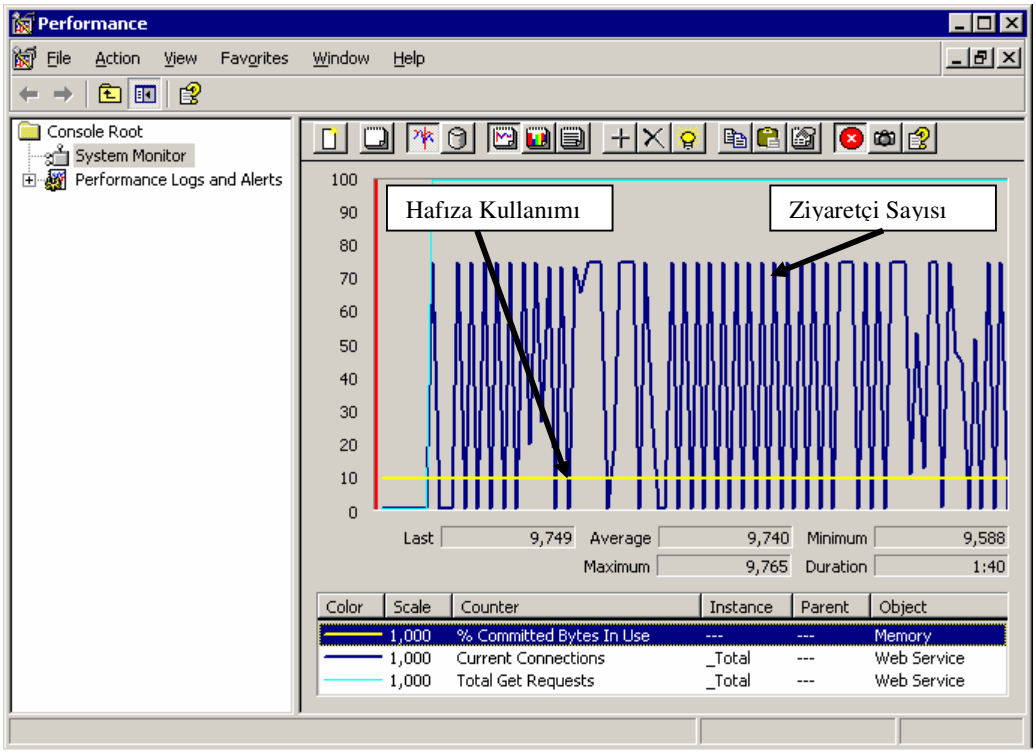
Şekil 4.6 Sayfa1.html işlemci kullanıcı grafiği

Şekil 4.6 da görüldüğü gibi mavi çizgi kullanıcı anlık sayısını kırmızı çizgi ise işlemci kullanım oranını göstermektedir. Statik sayfa1.html 75 kullanıcı tarafından 3 dakika boyunca 16950 kez sunucudan talep edilmiştir. Sunucu bu talepleri karşılayabilmek için maksimum %6.25, ortalama % 1.112 işlemci kullanımı ile çalışmıştır. Bunun nedeni statik sayfaların web sunucu tarafından sabit diskten okunarak sunucu tarafından istemci tarafına göndermesinden kaynaklanmaktadır.

Saniyedeki cevaplayabileceği istek sayısı ise;

Saniyedeki cevaplayabileceği istek sayısı = Toplam istek / Toplam süre eşitliğinden;

İstek sayısı = 94.16 istek/sn bulunur.

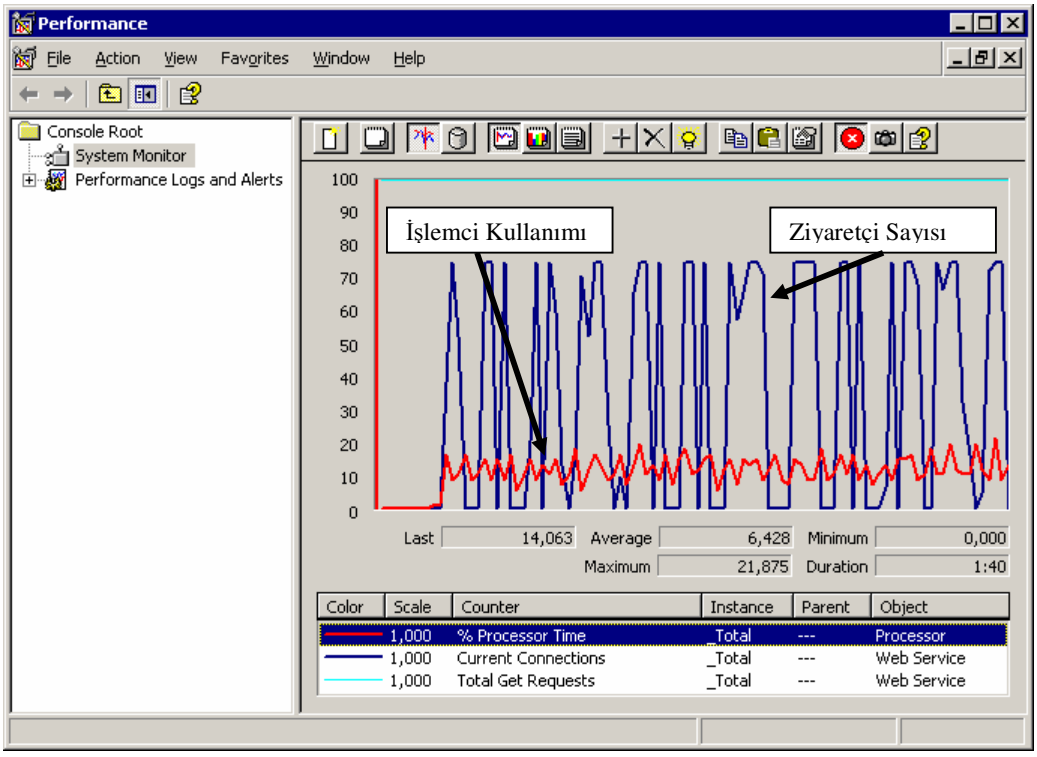


Şekil 4.7 Sayfa1.html hafıza kullanıcı grafiği

Şekil 4.7 deki grafikten ise 16950 talep sonrası hafızadaki maksimum artış yüzdesi % 0.177 olarak görülmektedir. Artış oranın bu denli olmasının nedeni statik sayfalarda web sunucunun fazla sistem kaynak ihtiyacı olmamasından kaynaklanmaktadır.

4.5 Test 4.2

İkinci testimizde ise Tablo 4.1 de açıklanan sayfa2.aspx dinamik öğeler içeren web uygulaması test edilmiş ve test anında oluşan kullanım istatistikleri grafiksel olarak gösterilmiştir.



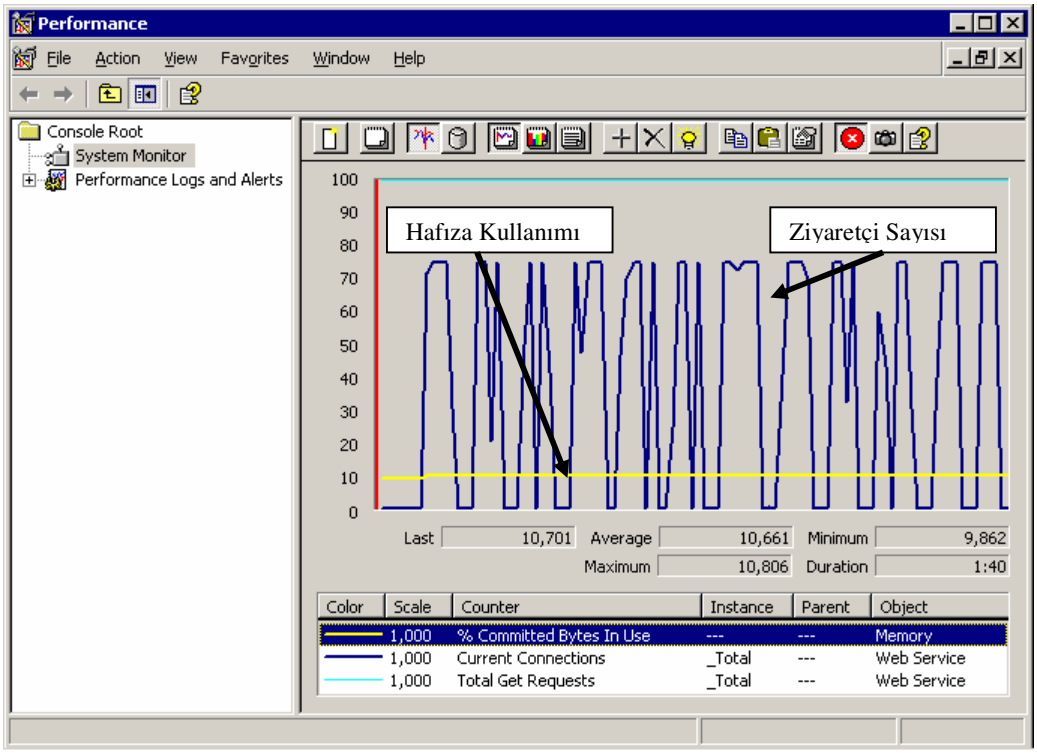
Şekil 4.8 Sayfa2.aspx işlemci kullanıcı grafiği

Şekil 4.8 dinamik öğeler içeren sayfa2.aspx 75 kullanıcı tarafından 3 dakika boyunca 9525 kez sunucudan talep edilmiştir. Sunucu bu talepleri karşılayabilmek için maksimum %21.875, ortalama % 6.428 işlemci kullanımı ile çalışmıştır. Bunun nedeni dinamik sayfaların web sunucu tarafında işlem yapmasına bağlı olarak işlemci kullanmasıdır.

Saniyedeki cevaplayabileceği istek sayısı ise;

Saniyedeki cevaplayabileceği istek sayısı = Toplam istek / Toplam süre eşitliğinden;

İstek sayısı = 52.91 istek/sn bulunur.

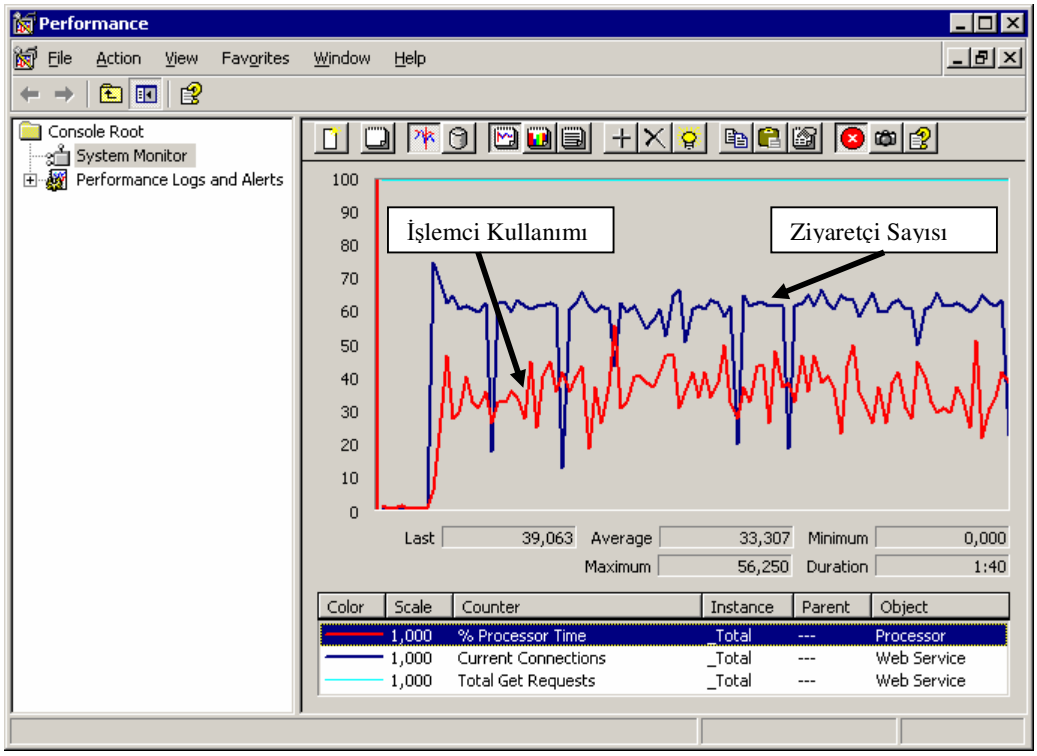


Şekil 4.9 Sayfa2.aspx hafıza kullanıcı grafiği

Şekil 4.9 deki grafikten ise 9525 talep sonrası hafızadaki maksimum artış yüzdesi % 0.944 olarak görülmektedir. Artış oranın bu denli olmasının nedeni dinamik sayfalarda web sunucunun sistem kaynaklarını talepleri karşılamak ve işlem yapma ihtiyacı olmasından kaynaklanmaktadır.

4.6 Test 4.3

Üçüncü testimizde ise Tablo 4.1'de açıklanan sayfa3.aspx web uygulamasının sistem kaynakların kullanımı ve talep edilme sıklığı sayaçlarının değerleri belirlenmeye çalışıldı. Sayfa3.aspx web uygulaması dinamik öğeler kullanılan ve veritabanı sunucusuna bağlanıp sorgulamalar sonucunda veri döndüren programlama yapısındadır.



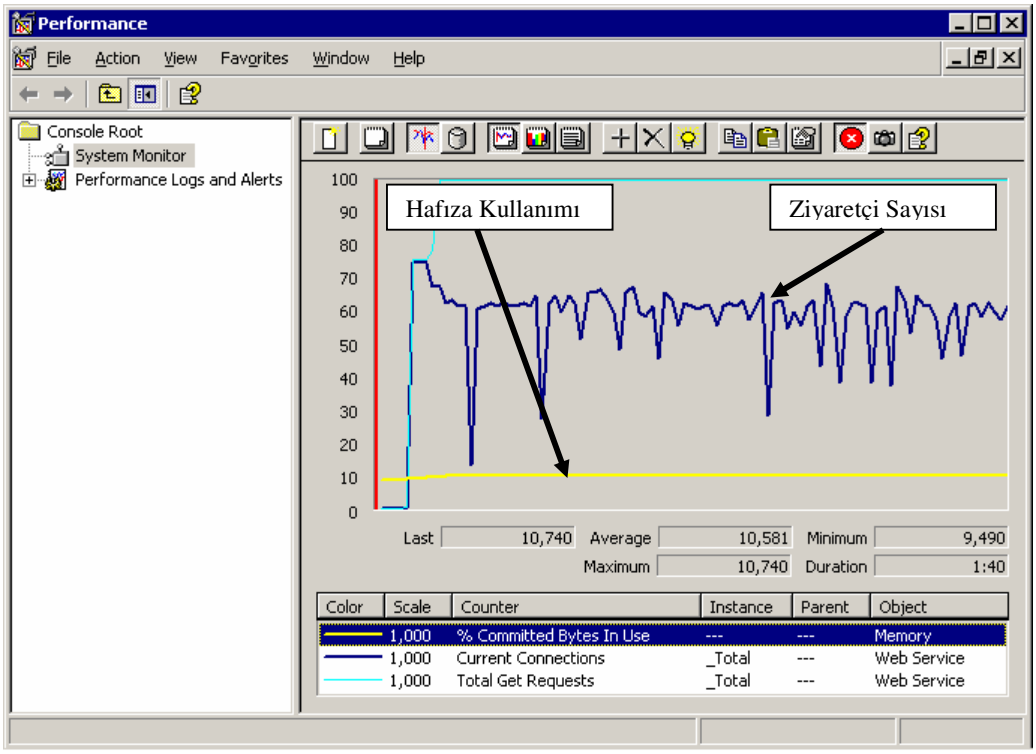
Şekil 4.10 Sayfa3.aspx işlemci kullanıcı grafiği

Şekil 4.10 dinamik öğeler içeren sayfa3.aspx 75 kullanıcı tarafından 3 dakika boyunca 3653 kez sunucudan talep edilmiştir. Sunucu bu talepleri karşılayabilmek için maksimum %56.250, ortalama % 33.307 işlemci kullanımı ile çalışmıştır. Bunun nedeni dinamik veritabanı bağlantılı sayfaların web sunucu tarafında işlem yapmasına bağlı olarak işlemciye ihtiyaç duymasındandır.

Saniyedeki cevaplayabileceği istek sayısı ise;

Saniyedeki cevaplayabileceği istek sayısı = Toplam istek / Toplam süre eşitliğinden;

İstek sayısı = 20.29 istek/sn bulunur.



Şekil 4.11 Sayfa3.aspx hafıza kullanıcı grafiği

Şekil 4.11 ise sayfa3.aspx e ait hafıza kullanım oranı gösterilmektedir. Buradan hafıza oranındaki artış %1.25 olarak hesaplanır. Bunun nedeni ise *oracle client* ve web sunucunun hafızaya daha fazla ihtiyaç duymasındandır.

Tablo 4.1 de bulunan diğer sayfalar sayfa3.html, sayfa4.aspx, sayfa5.aspx saf bayes tahmin modelinde kullanılmak üzere eğitim verileri bu şekilde oluşturulmuştur.

4.7 Testlerin Sonuçları

Performans testi sonucunda elde edilen veriler aşağıda özetlenmiştir. Performans niteliğinin sınıflarını belirlemek için rasgele 10 kullanıcının web sayfasından yapmak istediklerine ve beklentilerine karşı performansını değerlendirmesini 'kötü','normal','iyi' olarak istenmiştir. Bir sayfa için verilen cevapların çoğunluğu o uygulamanın performans niteliğinin sınıfı olarak kabul edilmiştir.

Uygulama	Sayfa1.html	Sayfa2aspx	Sayfa3.aspx	Sayfa4.aspx	Sayfa5.aspx	Sayfa6.aspx
İşlemci%	1,12	6,42	33,307	3,57	8,83	45,32
Hafıza%	0,177	0,944	1,25	0,273	1,33	1,45
Boyut(KB)	2,2	3,5	16	10	7,5	25
İstek/Sn	94,16	52,91	20,29	80,02	43,21	17,15
Dinamik	Yok	Var	Var	Yok	Var	Var
Veritabanı	Yok	Yok	Var	Yok	Yok	Var
Performans	İyi	Kötü	Normal	Normal	İyi	Kötü

Tablo 4.3 Saf Bayes Eğitim Tablosu

Tablo 4.3 teki verilere göre niteliklerin her birinin farklı sayaç değerleri elde edilmiştir. Saf Bayes uygulaması yapılabilmesi için bu sayaç değerleri indirgemeye uğratılması gruplanması sonucun hesaplanmasını ve eğitim kümesinin büyümesini sağlayacaktır. Bu tanımlamaları elimizdeki örneklere bakarak en mantıklı aralık veya tanımlayıcıların seçilmesi ile yapılmıştır.

Bu nedenle;

İşlemci % değerleri : ‘<5’, ‘5…25’, ‘>25’ Aralık değerleriyle tanımlanmıştır.
Hafıza % artış değerleri : ‘<0,5’, ‘0,5…1’, ‘>1’ Aralık değerleriyle tanımlanmıştır.
Boyut(Kb) : <7,5 ‘küçük’, 7,5…10 ‘orta’, >10 ‘büyük’ değerleriyle tanımlanmıştır.
İstek/Sn : >70 ‘çok’, 25…70 ‘orta’, <25 ‘az’ değerleriyle tanımlanmıştır.
Dinamik öğe : ‘var’, ‘yok’ değerleriyle tanımlanmıştır.
Veritabanı bağlantısı : ‘var’, ‘yok’ değerleriyle tanımlanmıştır.

Uygulama	Sayfa1.html	Sayfa2aspx	Sayfa3.aspx	Sayfa4.aspx	Sayfa5.aspx	Sayfa6.aspx
İşlemci%	<5	5...25	>25	<5	5…25	>25
Hafıza%	<0,5	0,5…1	>1	<0,5	>1	>1
Boyut(KB)	Küçük	Küçük	Büyük	Orta	Orta	Büyük
İstek/Sn	Çok	Orta	Az	Çok	Orta	Az
Dinamik	Yok	Var	Var	Yok	Var	Var
Veritabanı	Yok	Yok	Var	Yok	Yok	Var
Performans	İyi	Kötü	Normal	Normal	Normal	Kötü

Tablo 4.4 Veri İndirgenmiş Saf Bayes Eğitim Kümesi

4.8 Sınıflama Kurallarının Tanımı

Saf Bayes tahmin yöntemine göre elde edilen bu veriler eşliğinde her sütun için sınıflama kuralları tanımlanabilir. Buna örnek olarak aşağıdaki kural gösterilebilir.

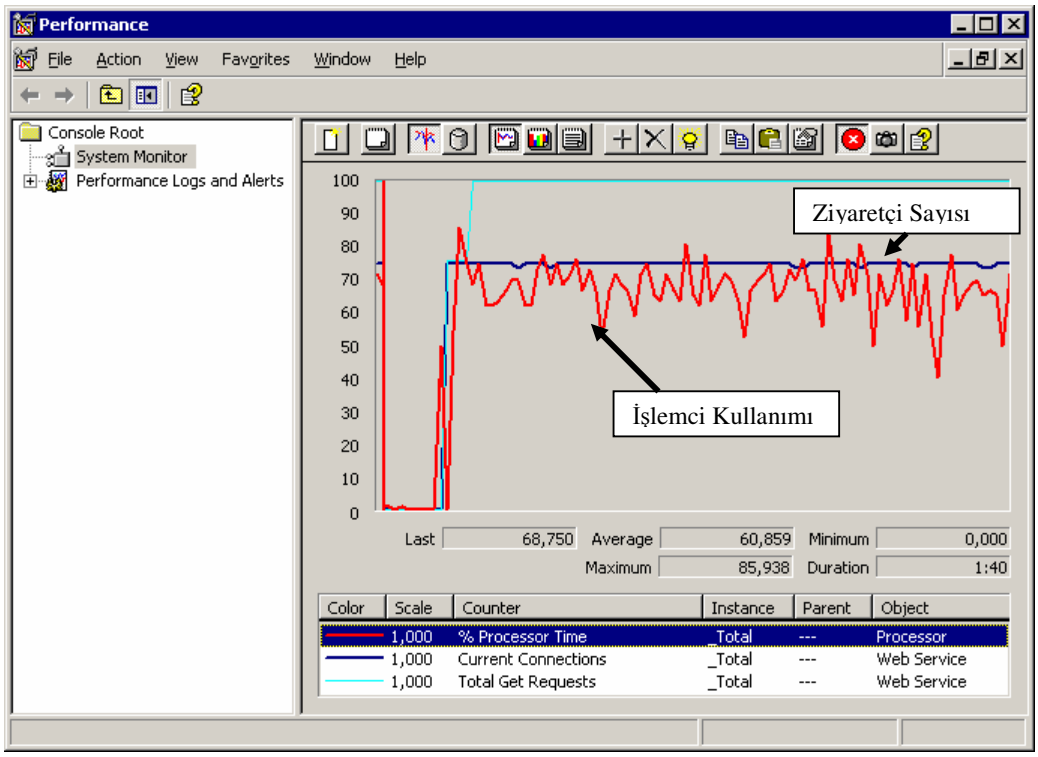
Eğer işlemci kullanımı '%5 ten küçük' ve hafıza gereksinimi '%0.5 ten az' ve sunucu tarafından 'saniyede cevaplama çok' ve 'dinamik öğe yok' ve 'veritabanı bağlantısı yok' ise web uygulamasının performans iyidir.

4.9 Saf Bayes Algoritmasının Uygulanması

Bayes sınıflama kuralları sonucu elimizde bulunan web sunucu limitlerini genel hatları ile çizmeye çalıştık. Buradan anlaşıldığı üzere yapılan farklı web programlama teknikleriyle ve web sayfasının sunucu kaynaklarını ne derecede kullandığını bilirsek yeni programlanan web uygulamasının kullanıcılar tarafından ne derece kullanışlı, ne kadar performanslı ya da ne kadar beğenilebileceği hakkında bir kestirim, tahmin yapabiliriz.

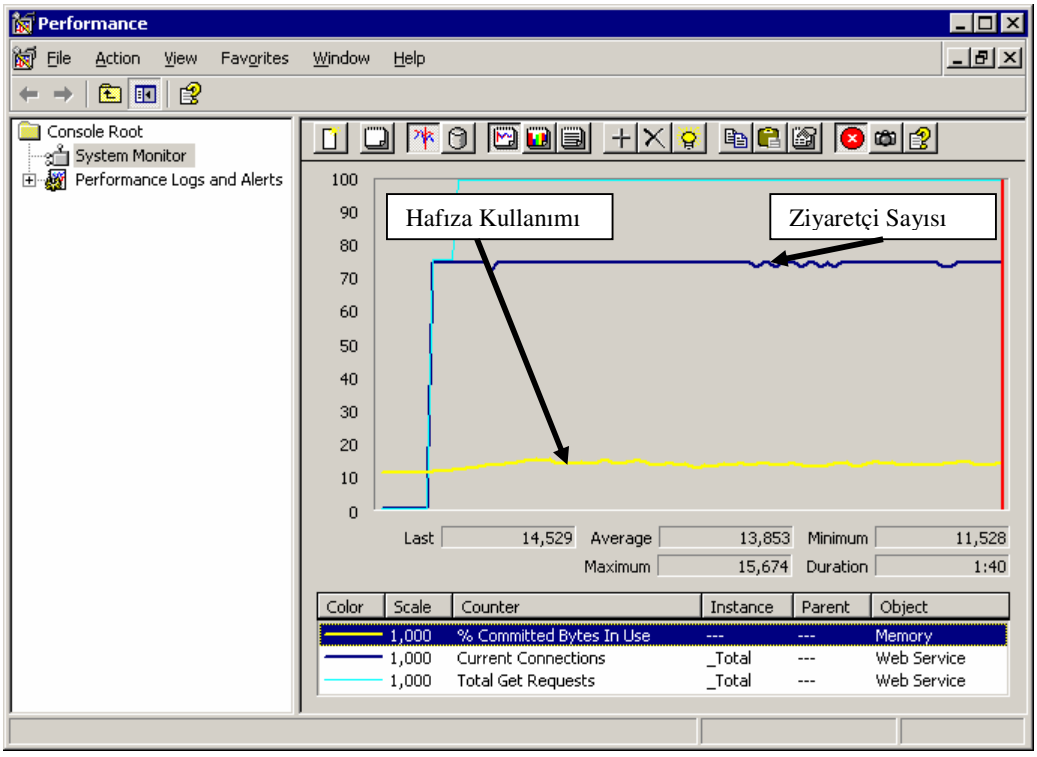
Oluşturulan sınıflama kurallarının güvenilirliğini ölçmek için yeni bir web uygulaması tasarlayabiliriz. Bu web uygulaması(test.aspx) basit bir mesajlaşma sayfasıdır. Öğrenci kendisine sınıf arkadaşları, ders hocası, bölüm başkanı tarafından gönderilen mesajları okuyabileceği ve cevaplayabileceği ya da yeni mesajlar yazabileceği bir web uygulaması örneği, test sayfasıdır. Bu web uygulaması dinamik öğelerle(ASP.NET) ile geliştirilmiş ve Oracle veritabanına bağlanan yapıya sahiptir. Bu sayfanın boyutu 45 KB' dır.

Test.aspx sayfasını Was ile 56 kbps bant genişliğine sahip 75 kullanıcı ile 3 dakika web sunucudan talep ederek işlemci, hafıza yüzdesi ve saniyedeki istek sayısını tespit etmemiz gerekir.



Şekil 4.12 Test.aspx işlemci kullanıcı grafiği

Şekil 4.12 de görüldüğü gibi işlemci kullanım yüzdesi ortalama 60,859' dır. Bunun nedeni ASP.NET in Oracle veritabanından öğrenciye ait mesajları sorgulaması ve bu mesajları web sayfasında düzenli bir şekilde hazırlamasından kaynaklanmaktadır.



Şekil 4.12 Test.aspx hafıza kullanıcı grafiği

Şekil 4.12 Test.aspx mesajlaşma sayfasının kullanılan hafıza artış miktarını göstermektedir. Bu değer görüldüğü gibi % 4,146 dir. Uygulama 3 dakika boyunca 4612 kez sunucudan talep edilmiştir.

Saniyedeki cevaplayabileceği istek sayısı ise;

Saniyedeki cevaplayabileceği istek sayısı = Toplam istek / Toplam süre eşitliğinden;

İstek sayısı = 25.62 istek/sn bulunur.

Buna göre X yani bilinmeyen performans tahmini problemini bölüm 3 te anlatılan Saf Bayes Algoritması ile çözersek;

X = (İşlemci % = ’>25’, Hafıza % = ‘>1’, Boyut = ‘büyük’, İstek/sn = ‘orta’, Dinamik = ‘var’, Veritabanı = ‘var’).

Şimdi Saf Bayes algoritması gereği P(X|Ci)P(Ci) maksimum yapan değeri bulmamız gerekir, Performans niteliğinin 3 farklı sınıf [kötü, normal, iyi] olduğu için i=1,2,3 için P(Ci) temel sınıf olasılıklarını hesaplarsak:

P(Performans= ‘kötü’) = 2/6 = 0,333
P(Performans= ‘normal’) = 3/6 = 0,5
P(Performans= ‘iyi’) = 1/6 = 0,166

Şimdi de P(X|Ci), i=1,2,3 hesaplamak için aşağıdaki işlemleri yapmamız gerekir.

P(işlemci%=’>25’| performans= ‘kötü’) = 1/6 = 0,166
P(işlemci%=’>25’| performans= ‘normal’) = 1/6 = 0,166
P(işlemci%=’>25’| performans= ‘iyi’) = 0/6 = 0

P(hafıza%=’>1’| performans= ‘kötü’) = 1/6 = 0,166
P(hafıza%=’>1’| performans= ‘normal’) = 2/6 = 0,333
P(hafıza%=’>1’| performans= ‘iyi’) = 0/6 = 0

P(boyut=’büyük’| performans= ‘kötü’) = 1/6 = 0,166
P(boyut=’büyük’| performans= ‘normal’) = 1/6 = 0,166
P(boyut=’büyük’| performans= ‘iyi’) = 0/6 = 0

P(istek=’orta’| performans= ‘kötü’) = 1/6 = 0,166
P(istek=’orta’| performans= ‘normal’) = 1/6 = 0,166
P(istek=’orta’| performans= ‘iyi’) = 0/6 = 0

P(dinamik=’var’| performans= ‘kötü’) = 2/6 = 0,333
P(dinamik=’var’| performans= ‘normal’) = 2/6 = 0,333
P(dinamik=’var’| performans= ‘iyi’) = 0/6 = 0

P(veritabanı=’var’| performans= ‘kötü’) = 1/6 = 0,166
P(veritabanı=’var’| performans= ‘normal’) = 1/6 = 0,166
P(veritabanı=’var’| performans= ‘iyi’) = 0/6 = 0

Yukarıdaki ihtimalleri kullanarak, elimizde

P(X|performans = ‘kötü’) = 0,166 x 0,166 x 0,166 x 0,166 x 0,333 x 0,166
= 0,27639
P(X|performans = ‘normal’) = 0,166 x 0,333 x 0,166 x 0,166 x 0,333 x 0,166
= 0,442224
P(X|performans = ‘iyi’) = 0 x 0 x 0 x 0 x 0 x 0
= 0

P(X|performans = ‘kötü’) P(Performans= ‘kötü’) = 0,27639 x 0,333
= 0,092120787
P(X|performans = ‘normal’) P(Performans= ‘normal’) = 0,442224 x 0,5
= 0,221112
P(X|performans = ‘iyi’) P(Performans= ‘iyi’) = 0 x 0,166
= 0

Bu hesapların sonucunda Saf Bayes Sınıflayıcısı örnek X(test.aspx) için Performans=’normal’ olarak tahmin eder, sınıflar.

5. SONUÇ VE ÖNERİLER

Web uygulamalarında istenen sayfaların hızlı ve hatasız bir şekilde görüntülenmek istenmektedir, kullanılan farklı ama neticesinde aynı işi yapabileceğimiz teknolojiler yardımıyla önemli olan istenen veriyi minimum süre içerisinde kullanıcıya iletmektir. Bunun içinde elimizde var olan bir sistemin kapasitesini ve uygulamada yapılacak değişikliklerin çalışan sistemi bozmadan bize ne kadar fayda ve rahatlık sağlayacağını bilmek ön plana çıkmaktadır.

Dinamik web uygulamaları yapıları gereği, statik sayfalara oranla bilgisayar sistemlerinin kaynaklarını daha fazla kullanırlar. Bunun neticesinde dinamik web uygulamaları az kullanmak ya da kullanmamak söz konusu olmamalıdır. Bunun yerine kullanıcıların ziyaret ettiği web sayfalarına küçük anketler koyarak web uygulamasından beklentilerinin karşılanıp karşılanmadığı sorulabilir. Böylelikle Saf Bayes tahmin ve sınıflandırma algoritması için eğitim küme setleri güncel tutularak web sunucu, web uygulaması ve ziyaretçi profili için tahminler yapılabilir ve performans için iyileştirmeler yapılmasına olanak tanınır.

Saf Bayes sınıflandırma algoritması performans analizi konularında sıkça kullanılmaktadır. Burada yapılan çalışma uygulamaların sunum ve konfigürasyonlarının daha iyi duruma getirebilmesi için yapılmıştır. Web uygulamasının programlama yönteminde değişiklik yapılarak performansın optimum hale getirmenin yanı sıra sistem kaynakları ile yapılacak iyileştirmeler sonucu da istenen verim elde edilebilir. Aşağıda sıralanan bazı değişiklikler ve ayarlamalar buna örnektir.

- **Daha iyi işletim sistemlerini tercih etmek:** Bellek yönetimi, işlem yönetimi, işlemci kullanımı daha iyi olan işletim sistemlerini kullanmak her açıdan avantaj sağlayacaktır.

- **Daha güçlü işlemcileri tercih etmek:** Birim zamanda yapılacak iş sayısı artırılarak bir uygulamanın daha hızlı çalışmasını sağlar.

- **Sistemdeki işlemci sayısı artırmak:** Aynı anda birden çok işlem yapılmasını sağlayarak performansta iyileştirme elde edilebilir.

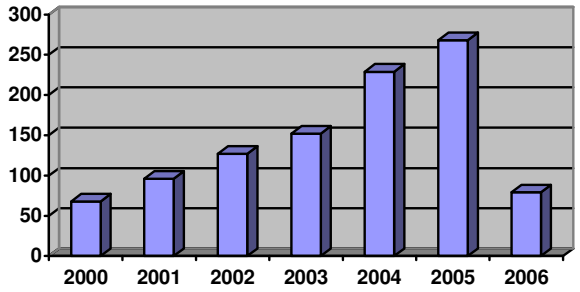
- **Sistemin Belleğini artırmak:** İşletim sistemi tarafından fiziksel bellek alanın yetersiz geldiği zamanlarda uygulamaların ihtiyacı olan bellek alanı sabit disk(Sanal Bellek) üzerinden karşılanır. Bu ise performansta kayıplar oluşturur.

- **Sabit disk kullanımında dönme ve okuma kafası hareketi hızlı olan modeller tercih etmek:** Sabit diskin bellek alanı olarak kullanılması ve web sayfalarının fiziksel bellekte tutulmasından dolayı veriye erişim mümkün olduğunca hızlı tutulmalıdır.

- **Sunucunun yerel ağa ulaşma hızı:** Sunucu bilgisayarın hizmet verdiği bant genişliği ne kadar büyük olursa veri transferindeki gecikmeler ve beklemeler azalır.

Yapılan çalışmayla her geçen gün artan web sayfalarını ziyaret eden kullanıcıların bu sayfaları ziyaret etmeleri sonucunda edindikleri tecrübe ve deneyimi bizim sistemimize aktarmaktadırlar. Bu aktarmayı ‘iyi’,’normal’,’kötü’ şeklinde üç sınıflamaya göre yapmaktadırlar. Bunun neticesinde elimizde bulunan web uygulamalarının ya da yeni uygulamaların nasıl bir sınıfa ait olduğunu tahmin etmemiz mümkün olmaktadır.

Web sayfalarının performans ölçümlerinde ve performans tahmininde veri madenciliği yöntemleriyle yapılması popüler bir konu olmaya devam etmektedir. Bu amaçla Selçuk üniversitesinin abone olduğu *sciencedirect, engineeringvillage2, ieeexplore.ieee* veritabanları üzerinde yıllara göre 'Mining Performance Prediction' kelimeleri taranmıştır. Buna göre elde edilen sonuçlar şekil 4.13'te gösterilmiştir.



Şekil 4.13 'Mining Performance Prediction' veritabanlarındaki makale sayısı

Tez yazılırken 2006 yılının tamamlanmamış olmasından dolayı 2006 yılına ait makale sayısı eksik çıkması bundan kaynaklanmış olabilir.

Performans tahmini ya da web sayfalarını ziyaret ederken kullanıcıların davranışlarının incelendiği çalışma[3] da en iyi sırasal web erişimi genetik algoritmalar ile incelenmiş fakat tahmin, kestirim yapılmamıştır. Bir başka çalışma da dinamik yapılı web programlama yapıları kıyaslanmış fakat kullanıcıların beğenisine göre değerlendirilmemiş ve veri madenciliği uygulanmamıştır[1]. Diğer bir web site sınıflama çalışmasında[2] ise web siteleri yapı, içerik ve erişim sayılarına göre hangi konuyla ilgili olduğu karar ağaçlarıyla sınıflandırma yapılmıştır. Bu uygulama da ise performans analizi eksik kalan bir konu olarak görülmektedir.

6. KAYNAKLAR

1. (Apte, V) Performance comparison of dynamic web platforms, Computer Communications 26 (2003) 888–898, Elsavier

2. (Estruch V.) Web Categorisation Using Distance-Based Decision Trees, Electronic Notes in Theoretical Computer Science 157 (2006) 35–40, Elsavier.

3. (Tug E.) Automatic discovery of the sequential accesses from web log data files via a genetic algorithm, Knowledge-Based Systems 19 (2006) 180–186, Elsavier.

4. (Bai G.), Performance benchmarking of wireless Web servers, accepted 10 January 2006, Networks xxx (2006) xxx–xxx, Elsevier

5. (Özekes S.)Veri Madenciliği Modelleri ve Uygulama Alanları, 2003, s.3, Haziran - Sayfa: 65-82, İstanbul Ticaret Üniversitesi Dergisi

6. Karadoğan, İ., Web Tabanlı Öğretim Sistemlerinin Kullandığı Sunucuların Performans Optimizasyonu, Yüksek Lisans Tezi, Sakarya.Ü., 2003

7. (Zhen Liu), Traffic model and performance evaluation of Web servers, Performance Evaluation 46 (2001) 77–100, Elsevier

8. (Mark S. Squillante), Web Traffic Modeling and Web Server Performance Analysis, Volume 5, 7-10 Dec. 1999 Page(s):4432 - 4439 vol.5, IEEE

9. Ji-Tzay Yang, Jiun-Long Huang, Feng-Jian Wang, A Tool Set to Support Web Application Testing, Department of Computer Science and Information Engineering National Chiao-Tung University, Taiwan, International Computer Symposium (ICS), October 1998

10. (John D illey), Web server performance measurement and modeling techniques, Performance Evaluation 33 (1998) 5-26, Elsevier

11. (Y. B. Lee), Design and Performance Evaluation of a Multimedia Web Server, journal of vısual communıcatıon and ımage representatıon Vol. 9, No. 3, September, pp. 183–193, 1998, article no: vc980395

12. (Iyengar A), An Analysis of Web Server Performance, Volume 3, 3-8 Nov. 1997 Page(s):1943 - 1947 vol.3, IEEE

13. The Art and Science of Web Server Tuning with Internet Information Services 5.0, George Reilly, Microsoft Corporation.

14. G.GÜVEN ders notları, http://iibf.erciyes.edu.tr/gg/veri/internetin_tanitimi.pdf

15. Kürşat CAGILTAY, Herkes icin internet, http://bid.ankara.edu.tr/start/hii/

16. Active Server Pages kitabı, Dr. Hakkı Öcal.

17. Tarık Özkan Gökay, Burak Akkuş, asp.net yazısı, http://www.msakademik.net/makaleler_detay.aspx?id=27.

18. Data Mining:Concept and Techniques Jiawei HAN kitabı, Nevcihan DURU ders notları

19. Sara Sprenkle, Automated Replay and Failure Detection for Web Applications, Computer and Information Sciences University of Delaware
http://128.4.133.74:8080/dspace/bitstream/123456789/87/3/sprenkle.ase05.pdf

20. Khalid Anwar, Arif Saleem, Web Application Stress Test and Data Analysis, http://www.khatme-nubuwwat.org/Stress.pdf