**Multi Thread Kullanarak Karar Ağacı Oluşturma**

*Bedirhan Şişman 130202036 Okan Demirel 130202015*

BilgisayarMühendisliğiBölümü

KocaeliÜniversitesi

[demirelokan1995@gmail.com](mailto:demirelokan1995@gmail.com) , [bedirhan.sisman@gmail.com](mailto:bedirhan.sisman@gmail.com)

**1. Projenin Özeti**

"Hasta" adında bir model oluşturuldu. haberman.txt dosyasındaki niteliklere göre Hasta modeline sırayla Yas,OperasyonYili,DugumSayisi(Hastalık Sayısı),Durum propert'leri oluşturuldu. Dosyadan okuma işlemleri ile birlikte dosyadaki nitelikler virgül ile ayrılırak(split methodu ile) veriler Hasta modeline aktarıldı. Böylelikle Hasta tipinde bir liste oluşturulmuş oldu. Hasta modelindeki Durum'un entropisi hesaplanması için, hastanın durum 1 olma olasılığı ve durum 2 olma olaslığı hesaplandı.Bu bilgiler ve entropi formülü ile hastanın durum entropisi hesaplandı. Diğer nitelikler için en iyi bölümleme değerini seçmek için, nitelikler için sırasıyla belirlenen bölümleme değerleri için kazanımlar hesaplandı. Her nitelik için en yüksek kazanımın olduğu bölümleme seçildi. Belirlenen en iyi bölümlemelere göre nitelikler için model üzerinde K(Kucuk), B(Buyuk) ayrımı yapıldı.Aynı zaman da en büyük kazanca sahip nitelik kök düğüm olarak belirlendi. Uç yaprakları bulmak için,kök düğüme göre diğer nitelikler için kazanç hesabı yaptıktan sonra,kök düğümün büyük ve küçük olmasına göre uç yapraklar belirlendi. Uç yaprakların durumları da 1 ve 2 olasılıklarına göre belirlendi. Sonuclar, Sonuc adında bir modele aktarılarak, windows form'da label'lara yazdırılarak ağaç çizdirildi.

**2. Projenin Amacı**

Bu projede bizden istenen multi thread yapısı kullanılarak verilen veri seti üzerinden karar ağacı oluşturulması istenmektedir. Karar ağacı oluşturma aşamasında C4.5 algoritmasının kullanılması istenmektedir. Projenin asıl amacı Thread yapısının kullanılması ve anlaşılmasıdır. Böylece eş zamanlı işlemyapılabilmektedir. Thread kullanımında 2 koşul istenmektedir. En iyi bölümlemenin elde edilmesi için sadece sayısal veriler üzerindethread kullanılarak işlem yapılması istenmektedir. Elde edilecek en iyi bölümleme durumu alt seviyede oluşacak düğüm sayısını verecektir. 2.istenen ise karar ağacında kök düğümden sonraki tüm seviyelerde oluşacak düğüm sayısı kadar thread oluşturulması ve her düğüm için bu threadlerin eş zamanlı olarak çalıştırılması istenmektedir. Bunlar dışında en iyi bölümleme için veri setinde 0-100 aralığında değerler alabilen x özelliği olduğunu kabul edilmiştir. Veri aralıklarını sırasıyla 10, 20 ve 50 olacak şekilde bölerek en fazla kazancı sağlayan aralığın elde edilmesi gerekmektedir. Kazanç hesabı entropi değerine göre yapılmalıdır. Burada 10, 20 ve 50 aralıkları için üç thread oluşturulmalı ve eş zamanlı entropi hesabının yapılması gerekmektedir. Düğüm sayıları için de : En iyi bölümleme sonucunda kazancı en yüksek aralık değerinin 20 oluğunu kabul edersek alt düğümün dalları şu şekilde olacaktır; x<20 , 20≤x<40 , 40≤x<60 , 60≤x<80 ve 80≤x<100. Daha sonra bu beş dal için veri setinde bulunan diğer bir özelliği ağaca eklemek için oluşan beş düğüme ait beş thread çalıştırılacaktır. Veri setinde yer alan tüm özellikler ağaca yerleştirilene kadar bu işlem devam edecektir. Ağacın her seviyesinde yer alacak özelliği seçerken yine entropi hesabının yapılması gerekmektedir**.** Karar ağacı öğrenmesi (decision tree learning) yöntemi, makine öğrenmesi (machine learning) konularından birisidir. Literatürde karar ağacı öğrenmesinin alt yöntemleri olarak kabul edilebilecek sınıflandırma ağacı (classification tree) veya ilkelleştirme ağacı (regression tree ,tahmin ağacı) gibi uygulamaları vardır. Karar ağacı öğrenmesinde, bir ağaç yapısı oluşturularak ağacın yaprakları seviyesinde sınıf etiketleri ve bu yapraklara giden ve başlangıçtan çıkan kollar ile de özellikler üzerindeki işlemeler ifade edilmektedir. Karar ağacı öğrenmesinde, ağacın öğrenilmesi sırasında, üzerinde eğitim yapılan küme, çeşitli özelliklere göre alt kümelere bölünür, bu işlem, özyineli olarak (recursive) tekrarlanır ve tekrarlama işleminin tahmin üzerinde bir etkisi kalmayana kadar sürer. Bu ağaçların çalıştırılması sırasında ID3 C4.5 algoritmaları gibi bazı algoritmalardan yararlanılabilir. Bu algoritmaların bazı avantajları vardır. Anlaşılması ve yorumlanması basittir. Basit bir açıklama ile karar ağacı öğrenmesinin neticelerini insanlar anlayabilirler. Hızlı ver ön işlemesi gerektirir: Çoğu alternatif tekniklere göre çok az bir işleme ile veri kullanılabilir hale gelir. Ön işleme aşaması diğer alternatiflerine göre daha kısa ve basittir. Hem sayısal hem de sınıfsal verilerin işlenmesi için kullanılabilir: Çoğu makine öğrenmesi algoritması, ya sayısal uygulamalarda kullanışlı ya da sınıflandırma problemleri için kullanışlıdır. Karar ağacı öğrenmeleri ise, iki alanda da kullanılabilir. Düşük hesaplama karmaşıklığına (computational complexity) sahiptir: Basit ve hızlı olmasından dolayı yüksek miktardaki veriyi kısa sürede işleyebilir ve alternatifi yöntemlere göre veri miktarı arttığında daha da tercih edilebilir olur.

**3. Yapılan Araştırmalar**

**3. 1Thread Yapısı**

Problem tanımında da bahsedildiği gibi projenin asıl amacı threadlerin kullanılmasıdır. Thread kavramı, aynı process ortamında birden fazla iş yürütme imkanı sağlar. Bir process’in çalışmaya başlaması ile birlikte bir thread oluşturulur ve bu process içerisinde birden multi-thread oluşturulabilir. Yaratılan iplikler aynı adres uzayında koşarlar. Her ipliğin kendine has program sayacı, saklayıcıları, yığını ve durumu(yok edilme, askıya alınma, önceliğinin değiştirilmesi) vardır. Bir thread’deki değişkeni aynı işlem tarafından üretilmiş bütün iplikler değiştirebilir veya okuyabilir.Threadler iş parçacıkları olup, çoklu görevlerde senkronizasyonu sağlamak için kullanılır. Her threadin farklı bir görevi vardır. Eğer bu threadler aynı anda aynı bölgeye girerse hatalar meydana gelebilir.Ek thread kullanımında, bilgisayarımızda bir programı çalıştırmak istediğimizde işlem bitene kadar programımızın arayüzü yanıt vermiyor olarak görünecektir. İşlem bitene kadar programımızdaki diğer kodlar çalışmaz. Bu durumların önüne geçebilmek için multithreading kullanılıyor. Multithreading de aynı anda birçok işlem yapabiliriz ve arayüzün yanıt vermemesine neden olmazlar.Threading geliştirme teknolojisiyle genellikle arka planda sürekli çalışmasını istediğimiz bir program parçacığı varsa kullanırız. Örneğin; bir antivirüs yazılımının gerçek zamanlı koruma yapabilmesi için thread teknolojisini kullanması gerekir.Thread kullanımı avantajlı bir kullanım gibi gözükse de yerinde ve etkili bir biçimde kullanılmayacak olursa, thread bizim program hızımızı arttırmak yerine programımızı yavaşlatır.

**3.2 C4.5 Ağacı**

Bu yazının amacı, [karar ağaçlarına (decision tree)](http://www.bilgisayarkavramlari.com/2012/04/07/karar-agaclari-decision-tree/) bir örnek olaran C4.5 ağacını açıklamaktır. C4.5 ağacı, [ID3 ağacının](http://www.bilgisayarkavramlari.com/2012/11/09/id3-tekrarli-ikilikci-agac-iterative-dichotomiser-3/) geliştirilmiş bir hali olarak düşünülebilir ve daha önce bu konuda yayınlanan [ID3 ağacı başlıklı yazıyı](http://www.bilgisayarkavramlari.com/2012/11/09/id3-tekrarli-ikilikci-agac-iterative-dichotomiser-3/) okumanızda yarar vardır. Bu yazıda iki ağacı karşılaştırarak konu anlatılacaktır.C4.5 ağacının [ID3 ağacından](http://www.bilgisayarkavramlari.com/2012/11/09/id3-tekrarli-ikilikci-agac-iterative-dichotomiser-3/) en büyük farkı [normalleştirme](http://www.bilgisayarkavramlari.com/2012/01/29/istatistiksel-normallestirme-statistical-normalisation/) kullanıyor olmasıdır. Yani ID3 ağacı üzerinde e [ntropi hesabı yapılır](http://www.bilgisayarkavramlari.com/2008/12/17/entropi-entropy/) (veya [bilgi kazanımı (information gain)](http://www.bilgisayarkavramlari.com/2012/11/13/information-gain-bilgi-kazanimi/)) ve bu değere göre karar noktaları belirlenir. C4.5 ağacında ise [entropi değerleri](http://www.bilgisayarkavramlari.com/2008/12/17/entropi-entropy/) birer oran olarak tutulur. Ayrıca ağaç üzerinde erişim sıklıklarına göre alt ağaçların (subtree) farklı seviyelere taşınması da mümkündür. C4.5 ağacının diğer bir farkı ise tam bu noktada orataya çıkar [ID3 ağacının](http://www.bilgisayarkavramlari.com/2012/11/09/id3-tekrarli-ikilikci-agac-iterative-dichotomiser-3/) yaklaşımından farklı olarak C4.5 [ağacında budama (prunning)](http://www.bilgisayarkavramlari.com/2009/11/23/alfa-beta-budamasi-alpha-beta-prunning/) işlemi yapılmaktadır.



*Şekil 1* :Bilgi Kazancı

Bilgi (information) hesaplaması sırasında kullanılacak olan formül yukarıdaki şekildedir. Buna göre herhangi bir örnek için o sınıftaki (S ile gösterilmiştir) değerlere göre frekansına bakılır. Ayrıca yukarıdaki formülde |M| değeri, o sınıftaki örneklerin sayısını ifade etmektedir. Yukarıdaki şekilde her örnek için bilgi değeri hesaplandıktan sonra kazanım (gain) hesaplanması mümkündür. Genelde bu adımda bilgi parçalara bölünür ve bölünen parçalar üzerinden işlem yapılır. Bu durum için ise hesaplama aşağıdaki şekilde yapılabilir:

*Şekil 2* : Her i parçası için bilgi hesaplaması



Şekil 2 deki formüüle göre her bir i parçası için yapılan bilgi hesaplaması verilmektedir. Kazanım ise bu durumda aşağıdaki şekilde hesaplanabilir

*Şekil 3* :Kazanım



Yani herhangi bir X özelliği için kazanım değeri, o özelliğin bağlı olduğu bütün parça ve sadece o özelliği ilgilendiren parça arasındaki farka eşittir.Projede information hesabı bu şekilde yapılmıştır.

**3.3 OOP SINIFI**

Projede kullanılması istenen bir diğer şey ise class yapısıdır. Böylece proje nesneye dayalı bir hale gelmektedir.OOP’ nin başlıca özelliği, yazılımda birimselliği (modularity) benimsemesidir. OOP ayrıca, bilgi gizleme (informationhiding), veri soyutlama (dataabstraction), çok biçimlilik (polymorphism) ve kalıtım (inheritance) gibi yazılımın bakımını ve aynı yazılım üzerinde birden fazla kişinin çalışmasını kolaylaştıran kavramları da yazılım literatürüne kazandırmıştır. Sağladığı bu avantajlardan dolayı, OOP günümüzde geniş çaplı yazılım projelerinde yaygın olarak kullanılmaktadır.OOP'nin altında yatan birimselliğin ana fikri, her bilgisayar programının , etkileşim içerisinde olan birimler veya nesneler kümesinden oluştuğu varsayımıdır. Bu nesnelerin her biri, kendi içerisinde veri işleyebilir, ve diğer nesneler ile çift yönlü veri alışverişinde bulunabilir. Hâlbuki OOP'den önce var olan tek yaklaşımda programlar sadece bir komut dizisi veya birer işlev kümesi olarak görülmektedirler. Günümüzde çok çeşitli nesne tabanlı programlama dilleri olmasıyla beraber, en popüler diller sınıflar üzerine kurulmuşlardır (class-based). Bu dillerde nesneler sınıfların birer üyesidir ve nesnelerin tipini de bu sınıflar belirlerler.En yaygın OOP dillerinden bazıları, Python, C++, Objective-C, Smalltalk, Delphi, Java, Swift, C#, Perl, Ruby ve PHP' dir.NYP'yi destekleyen programlama dilleri genellikle tekrar kullanım ve genişletilebilirlik açısından, prototipler ve sınıflar şeklinde kod kalıtımına sahiptirler ve NYP deki nesneler kimi zaman gerçek dünyada bulunan şeylere karşılık gelebilir. Mesela bir grafik programı "kare", "üçgen" ve "yuvarlak" gibi nesnelere sahip olabilir. Bazen nesneler daha soyut şeyleri de temsil edebilirler, mesela bir hesap makinesi programı logaritma almanızı ya da metreyi fite çevirmenizi sağlayacak nesnelere sahip olabilir. İşte bu bahsi geçen nesneler belli sınıflar altında bulunurlar, mesela "Kasım" adında bir nesne Aylar sınıfının bir üyesi olabilir ya da "Faktöriyel" olarak isimlendirilmiş bir nesne Fonksiyonlar sınıfında yer alıyor olabilir.Class yapısıHaberman.hheader dosyasının içinde oluşturulmuştur ve private ve public değişkenler tanımlanmıştır.Ayrıyeten projede dinamik dizi işlevi gören vektör yapısıda kullanılmıştır.Vektör (vector) standart isim uzayının (std::) bir parçasıdır ve aslında bir sınıf şablonudur. STL (Standart Şablon Kütüphanesi)’in bir üyesidir ve bir ardışık topluluk tur. Vektör bir dinamik dizi gerçeklemesidir, programlarda hemen hemen her tür veri tipi ya da nesneyi tutan bir dizi oluşturmak için kullanılabilir.Vektör eski C stili diziler yerine, verileri dinamik olarak bir dizide tutmamız gereken ve bu dizilerin otomatik olarak genişleyip küçülmesi gerektiği durumlarda ideal bir seçim olacaktır. Eski C stili diziler bunu yapamazlar. Bu projede vektör her yerde kullanılmıştır. İçiçe vektör yapısı kullanılmıştır ayrıca threadyapısıda vektör içine atılmıştır.

**4. Tasarım**

**4.1 Yazılım Dili**

Projeyi yazarken C# dilini kullandık.C# Programlama Dili (*si şarp* şeklinde telaffuz edilir), [Microsoft](https://tr.wikipedia.org/wiki/Microsoft)'un geliştirmiş olduğu yeni nesil programlama dilidir. Yine Microsoft tarafından geliştirilmiş [.NET Teknolojisi](https://tr.wikipedia.org/wiki/.NET_Framework) için geliştirilmiş dillerden biridir.Microsoft tarafından geliştirilmiş olsa da [ECMA](https://tr.wikipedia.org/wiki/ECMA_Standartlar_Birli%C4%9Fi) ve [ISO](https://tr.wikipedia.org/wiki/ISO) standartları altına alınmıştır.[C programlama dilinde](https://tr.wikipedia.org/wiki/C) bir tamsayı değişkeni 1 atırmak için ++ soneki kullanılır. [C++](https://tr.wikipedia.org/wiki/C%2B%2B) dili adını, C diliyle Nesneye Yönelimli Programlama yapabilmek için eklentiler (C With Classes) almıştır. Benzer şekilde C++ diline yeni eklentiler yapılarak ((C++)++) bir adım daha ileriye götürülmüş ve tamamen nesneye yönelik tasarlanmış C# dilinin isimlendirilmesinde, + karakterlerinin birbirlerine yakınlaşmış hali ve bir [melodi](https://tr.wikipedia.org/wiki/Melodi) anahtarı olan [C# Major](https://tr.wikipedia.org/wiki/Nota_(m%C3%BCzik)) kullanılmıştır.Bu dilin tasarlanmasına [Pascal](https://tr.wikipedia.org/wiki/Pascal_(programlama_dili)), [Delphi](https://tr.wikipedia.org/wiki/Delphi_(programlama_dili)) derleyicileri ve J++ programlama dilinin tasarımlarıyla bilinen [Anders Hejlsberg](https://tr.wikipedia.org/wiki/Anders_Hejlsberg) liderlik etmiştir.Birçok alanda [Java](https://tr.wikipedia.org/wiki/Java_programlama_dili)'yı kendisine örnek alır ve C# da java gibi C ve C++ kod sözdizimine benzer bir kod yapısındadır. .NET kütüphanelerini kullanmak amacıyla yazılan programların çalıştığı bilgisayarlarda uyumlu bir kütüphanenin ve yorumlayıcının bulunması gereklidir. Bu, Microsoft'un [.Net Framework](https://tr.wikipedia.org/w/index.php?title=.Net_Framework&action=edit&redlink=1)'u olabileceği gibi ECMA standartlarına uygun herhangi bir [kütüphane](https://tr.wikipedia.org/w/index.php?title=K%C3%BCt%C3%BCphane_(yaz%C4%B1l%C4%B1m)&action=edit&redlink=1) ve yorumlayıcı da olabilir. Yaygın diğer kütüphanelere örnek olarak [Portable.Net](https://tr.wikipedia.org/w/index.php?title=Portable.Net&action=edit&redlink=1) ve [Mono](https://tr.wikipedia.org/wiki/Mono) verilebilir.Özellikle nesne yönelimli programlama kavramının gelişmesine katkıda bulunan en aktif programlama dillerinden biridir .NET platformunun anadili olduğu bazı kesimler tarafından kabul görse de bazıları bunun doğru olmadığını savunur.C#, .NET orta seviyeli programlama dillerindendir. Yani hem makine diline hem de insan algısına eşit seviyededir. Buradaki orta ifadesi dilin gücünü değil makine dili ile günlük konuşma diline olan mesafesini göstermektedir. Örneğin; [Visual Basic](https://tr.wikipedia.org/wiki/Visual_Basic) .NET (VB.NET) yüksek seviyeli bir dildir dersek bu, dilin insanların günlük yaşantılarında konuşma biçimine yakın şekilde yazıldığını ifade etmektedir. Dolayısıyla VB.NET, C#.NET'ten daha güçlü bir dildir diyemeyiz. Programın çalışması istenen bilgisayarlarda framework kurulu olması gerekmektedir.

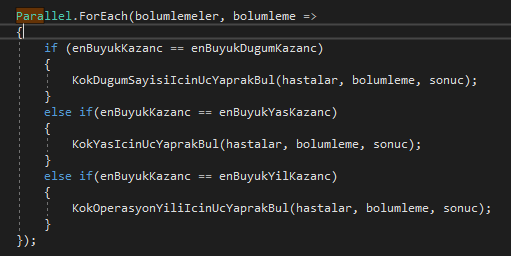
**4.2 Kullanılan IDE – Microsoft Visual Studio**

Microsoft Visual Studio, Microsoft tarafından geliştirilen bir tümleşik geliştirme ortamıdır (IDE). [Microsoft Windows](https://tr.wikipedia.org/wiki/Microsoft_Windows), [Windows Mobile](https://tr.wikipedia.org/wiki/Windows_Mobile), [Windows CE](https://tr.wikipedia.org/wiki/Windows_CE), [.NET Framework](https://tr.wikipedia.org/wiki/.NET_Framework), [.NET Compact Framework](https://tr.wikipedia.org/w/index.php?title=.NET_Compact_Framework&action=edit&redlink=1) ve [Microsoft Silverlight](https://tr.wikipedia.org/wiki/Microsoft_Silverlight) tarafından desteklenen tüm platformlar için yönetilen kod ile birlikte yerel kod ve Windows Forms uygulamaları, web siteleri, web uygulamaları ve web servisleri ile birlikte konsol ve grafiksel kullanıcı arayüzü uygulamaları geliştirmek için kullanılır.Visual Studio [IntelliSense](https://tr.wikipedia.org/w/index.php?title=IntelliSense&action=edit&redlink=1)'in yanı sıra "code refactoring" destekleyen bir kod editörü içerir. Entegre hata ayıklayıcı, hem kaynak-seviyesinde hem de makine-seviyesinde çalışır. Diğer yerleşik araçlar, [GUI](https://tr.wikipedia.org/wiki/GUI) uygulamaları, web tasarımcısı, sınıf tasarımcısı ve [veritabanı](https://tr.wikipedia.org/wiki/Veritaban%C4%B1) şema tasarımcısı yaratabilmek için bir form tasarımcısı içerir. Hemen hemen her işlevsellik düzeyinde dahil olmak üzere, kaynak kontrol sistemleri için destek (Subversion ve Visual SourceSafe gibi) sunan eklentileri kabul eder.Visual Studio, değişik programlama dillerini destekler, bu da kod editörü ve hata ayıklayıcısının neredeyse tüm programlama dillerini desteklemesini sağlamaktadır. Dahili diller [C](https://tr.wikipedia.org/wiki/C_(programlama_dili))/[C++](https://tr.wikipedia.org/wiki/C%2B%2B), [VB.NET](https://tr.wikipedia.org/wiki/Visual_Basic_.NET) (Visual Basic .NET üzerinden), [C#](https://tr.wikipedia.org/wiki/C_Sharp) (Visual C# ile), ve [F#](https://tr.wikipedia.org/wiki/F_Sharp) (Visual Studio 2010 itibarıyla) içermektedir.

Visual Studio özünde herhangi bir programlama dili, çözüm veya aracı desteklemeyerek, bunun yerine, bir VSPackage olarak kodlanmış işlevsellik sağlar. Yüklendiğinde, işlevsellik, *Hizmet* olarak kullanılabilir. IDE üç hizmet vermektedir; SVsSolution, projeler ve çözümleri numaralandırma yeteneği sağlar; SVsUIShell, pencereleme ve [UI](https://tr.wikipedia.org/wiki/Kullan%C4%B1c%C4%B1_aray%C3%BCz%C3%BC) işlevselliği (sekmeler, araç çubukları ve araç pencereleri dahil) sağlar; ve SVsShell, VSPackages kaydı ile ilgilenir. *Dil Hizmetleri* adı verilen belirli bir VSPackage kullanarak programlama dilleri için destek eklenmiştir. Bir dil hizmeti, çeşitli arayüzleri tanımlar, bu şekilde VSPackage uygulamaları, çeşitli fonksyonlara destek ekleneyebilir. Bu şekilde eklenebilir işlevler, sözdizimi renklendirme, deyim tamamlama, ayraç eşleşen, parametre bilgileri, üye listeleri ve arka plan derleme hata işaretleri olarak bulunmaktadır.Visual Studio, yerleşik herhangi bir kaynak denetimi desteği içermez ama IDE ile entegre etmek için kaynak kontrol sistemlerine iki alternatif yol tanımlar. Kaynak Kontrolü VSPackage kendi özelleştirilmiş kullanıcı arabirimini sağlayabilir. Buna karşılık, standart bir Visual Studio kullanıcı arayüzü ile MSSCCI (Microsoft Source Code Control Interface) kullanarak bir kaynak denetimi eklentisi, çeşitli kaynak denetimi işlevselliğini uygulamak için kullanılan işlevler kümesi sağlar.

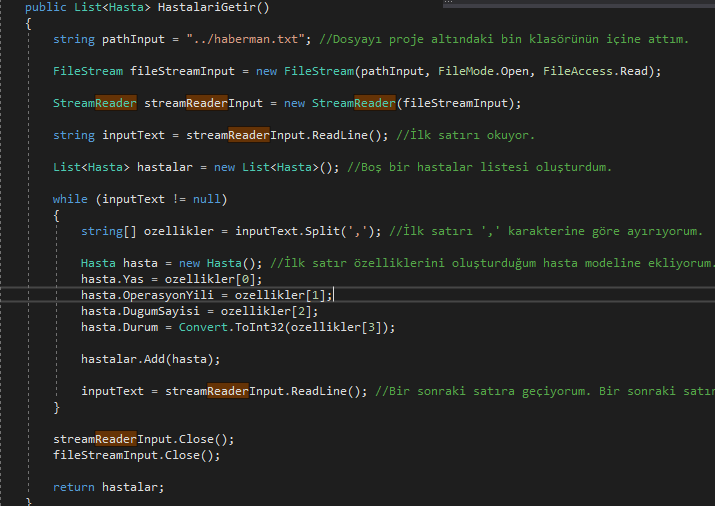
**4.3 Yazılım Mimarisi**

Paralellel.ForEach ile threadleri projeye entegre edildi.

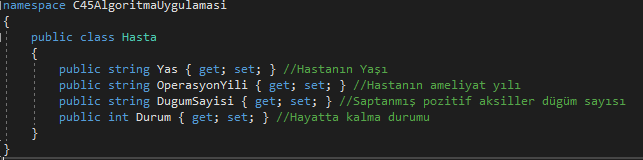


*Şekil 4:Parallel.ForEach*

Dosyadan okuma işlemi aşağıdaki şekildeki gibi yapıldı.

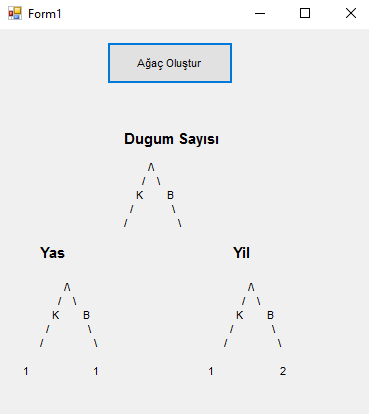
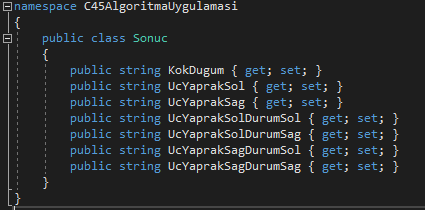


*Şekil 5:Dosyadan okumanın yapıldığı kod bloğu*



*Şekil 6:Parametrelerin bulunduğu sınıf*

*Şekil 7:Ağacı çizdirirken kullanılan sınıf ve elemanları*



*Şekil 8:Program çıktısı*

**5. Referanslar**

<http://bilgisayarkavramlari.sadievrenseker.com/2012/11/13/c4-5-agaci-c4-5-tree/>

<http://w3.gazi.edu.tr/~akcayol/files/WM_L4SupervisedLearning.pdf>

<https://aozsoyler.blogspot.com.tr/2011/04/karar-agaclar_30.html>

<http://www.ugurkizmaz.com/YazilimMakale-1720-C--Parallel-For---Parallel-Foreach.aspx>

<https://tr.wikipedia.org/wiki/C_Sharp>

#### Veri Madenciliği Ders Slaytları (Yrd.Doç.Dr. Sevinç İLHAN)