ISSN: 1309 – 6575
Eğitimde ve Psikolojide Ölçme ve Değerlendirme Dergisi
Journal of Measurement and Evaluation in Education and Psychology
2019; 10(1);80-95

Veri Madenciliğinde Kullanılan Bir Analiz Programı: WEKA

Gökhan AKSU* Nuri DOĞAN**

Öz

Bu çalışmada veri madenciliği yöntemlerinden ve bu alanda en çok kullanılan programlardan birini tanıtmak amaçlanmıştır. Bu amaç kapsamında WEKA programı ve bağımlı değişkenin bağımsız değişkenler tarafından tahmin edilmesi amacıyla kullanılan yöntemlerden biri olan karar ağaçları tanıtılacaktır. Teknoloji çağının yaşandığı günümüzde sahip olunan bilgi miktarı sürekli artış göstermekte ve bu bilgilerden anlamlı sonuçlar çıkarmak oldukça değerli bir çalışma alanı olarak görülmektedir. Veri madenciliğinde büyük miktardaki verinin içinde saklı olan ve araştırmacılar için oldukça faydalı olan bilgilerin bir dizi işlemlerin ardından ortaya çıkarılması hedeflenmektedir. Genel olarak tahminleme ve sınıflama üzerine kurulu bu yaklaşımda yeni ve doğruluğu henüz tam olarak test edilmemiş birçok yazılım bulunmaktadır. Bu çalışmada alan yazında veri madenciliği ile ilgili tarama yaptığınızda ilk karşınıza çıkan programlardan biri olan WEKA yazılımının ne olduğu, programın nasıl çalıştırıldığı ve analizler ile çıktı dosyalarının neler içerdiği açıklanmaya çalışılmıştır. Çalışmada ayrıca bu program üzerinden analiz yapmak isteyen uygulayıcılar için yazılımın üstün yönlerinin neler olduğu anlatılarak neler yapabilecekleri konusunda önerilerde bulunulmuştur.

Anahtar Kelimeler: Veri madenciliği, WEKA, sınıflama, tahmin, algoritma

GİRİŞ

Son on yılda, bilgiye ulaşım ve onu kullanabilme konusunda internet sayesinde büyük bir devrim yaşanmıştır (Jain, 2015). Bu aşamada araştırmacılar ve bilim insanları verilerin depolanması, geri çağrılması ve gerektiğinde kullanılması üzerine odaklanmışlardır. Zaman zaman eldeki bu veri belli bir alanda araştırma ve geliştirme yapmak için altın madenine benzetilmektedir. Veri madenciliği elde edilen veriyi girdi ve çıktı bilgisi olarak tanımlayan bir süreçtir (Weiss ve Davison, 2010). Alanda en çok atıf alan araştırmacılardan biri olan Fayyad, Piatetsky-Shapiro ve Smyth (1996) veri madenciliğini eldeki veriden belli örüntüleri ortaya çıkarmak için belirli algoritmaların uygulanması olarak tanımlamaktadır.

Veri seti üzerinde kolaylıkla gösterilebilecek şekilde basit yapıların farklı türleri oluşmaktadır. Örneğin bir veri setinde, tek bir özelliğin çok iyi çalıştığı diğerlerinin ilişkisiz veya gereksiz olduğu örnekler olabilir. Başka bir veri setinde, özelliklerin birbirinden bağımsız ve çıktı değişkenine eşit bir şekilde katkı sağladığı durumlar olabilir. Başka bir veri setinde basit bir mantıksal yapıya sahip ve karar ağacı tarafından elde edilen birkaç özelliğin yer aldığı durumlar olabilir. Farklı bir veri setinde, örnekleri farklı sınıflara atayan birkaç bağımsız kural kümesi olabilir. Bir başka bir veri setinde özelliklerin farklı alt kümeleri arasındaki birbirine bağımlı olma durumları incelenebilir. Daha farklı bir veri setinde uygun ağırlıklandırma yönetimiyle belirlenmiş sayısal özelliklerin ağırlıklı toplamları arasındaki doğrusal bağımlılık incelenebilirken bir diğerinde örneklerin birbirleri arasındaki mesafeleri esas alarak örnek uzayın belirli bölgelerine bu örnekleri yerleştirmek olabilir. Diğer bir veri setinde, öğrenmenin denetimsiz olduğu algoritmalar da görüldüğü gibi hiçbir sınıf değeri olmayabilir (Witten ve Frank, 2005). Olası veri setlerinin sonsuz eşitliliğinde farklı yapıların ve farklı veri madenciliği araçlarının kullanıldığı farklı yapıda örnekler bulunmaktadır. Bu örneklerde farklı bir türün uygunluğu tamamen gözden kaçırabilen ve sadece tek bir sınıfa yönelik doğru sınıflamalar yapan algoritmalar da bulunabilmektedir (Holte, 1993).

*Dr., Adnan Menderes University, Aydın Vocational School, Aydın-Turkey, e-mail: gokhanaksu1983@hotmail.com ORCID ID: 0000-0003-2563-6112
** Prof. Dr., Hacettepe University, Education Faculty, Ankara-Turkey, e-mail: nuridogan2004@gmail.com, ORCID ID: 0000-0001-6274-2016

Bu makaleye atıfta bulunmak için:
Aksu, G., & Doğan, N. (2010). An analysis program used in data mining: WEKA. *Eğitimde ve Psikolojide Ölçme ve Değerlendirme Dergisi, 10*(1), 80-95. DOI: 10.21031/epod.399832

Received: 28.02.2018
Accepted: 29.11.2018

Sınıflama kuralları karar ağaçlarının alternatifleridir. Verilen kurallar ilk önce birinci satır, sonra ikinci ve sırasıyla son satır olacak şekilde yorumlanmaktadır. Kuralların yer aldığı bu satırlar genelde "Karar Listesi olarak" isimlendirilmektedir. Bu listeler, karar tablosunda yer alan durumların tamamını doğru bir şekilde sınıflaması için gerekli kuralları göstermektedir (Chadha ve Singh, 2012).

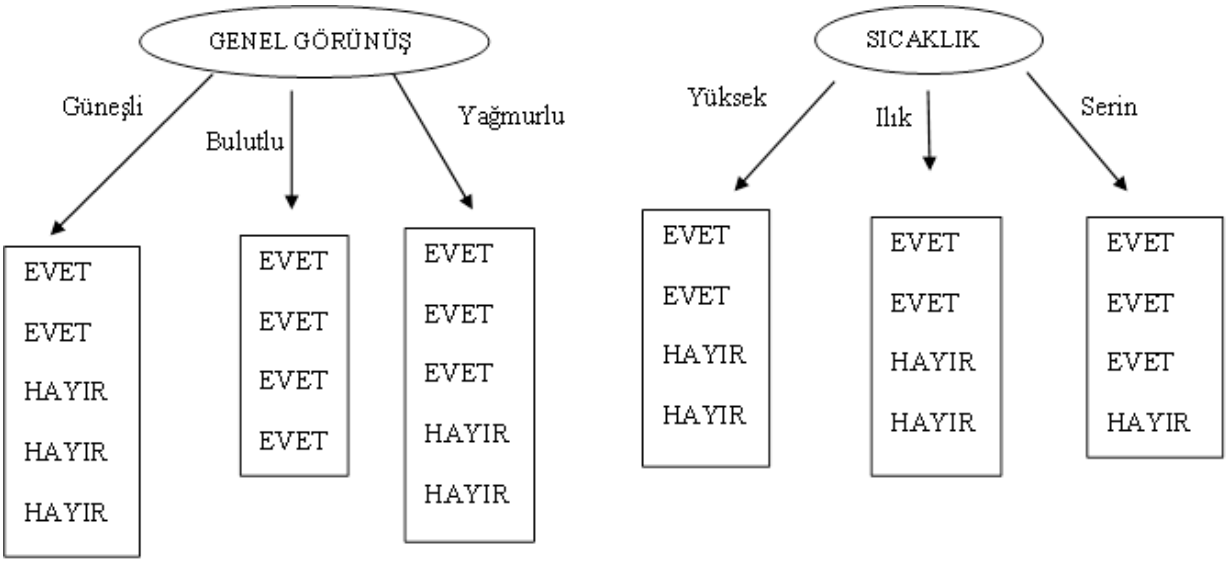
Birliktelik kuralları sadece sınıflama yapmayıp herhangi bir özelliği tahmin edebilmesi dışında sınıflama kurallarından farkı yoktur. Veri setindeki örüntülerin, ilişkilerin ve nedensel yapıların ortaya çıkarıldığı bu yaklaşımda farklı özellik kombinasyonları da tahmin edilebilmektedir. Bunun yanında, birliktelik kuralları sınıflama kurallarında olduğu gibi bir dizi kural kümesinin birlikte kullanılmasını da gerektirmezler (Han, Kamber ve Pei, 2000). Çok sayıda farklı ilişkilendirme kuralı küçük bir veri kümesinden bile türetilebilmektedir. Bu sebeple çok fazla örneğe uygulama amacı yoktur. Birliktelik kuralları genel olarak …if…then… döngüsü ile oluşturulmaktadır. Örneğin "eğer rüzgar yok ve oyun oynanmamış ise nem değeri yüksektir" birliktelik kuralı, rüzgar olmadığında oyun oynanmış ise nem değerinin yüksek olması gerektiğini belirtmektedir (Witten ve Frank, 2005).

Kümeleme kuraları veri setindeki örneklerin belirtilen kümelere nasıl girdiğini gösteren bir diyagram şeklini almaktadır. Küme aidiyetlerinin belirlendiği bu yöntemde elde edilen sonuçlar bazen dendogram bazen de tablo olarak verilmektedir. Veri setindeki bir örneğin sahip olduğu özelliklere göre diğer örnekler ile benzerlik ya da benzemezlik ölçütlerine bağlı olarak hangi kümeye ait olduğu gösterilmektedir. Dendogram üzerinde her küme bir alt seviyesinde kendisinin alt kümelerine ayrışmış olarak gösterilmektedir (Karypis, Han, ve Kumar, 1999).

Yordama işleminde kategorik değişkenler yerine sayısal özellikler ele alınmaktadır. Veri madenciliğinde her ne kadar karar ağaçları veya birliktelik kurallarında kategorik değişkenler ele alınsa da özelliğin sayısal değerini tahmin etmede doğrusal regresyon modellerinden yararlanılmaktadır (Padmavathi, 2012). Klasik regresyon yöntemlerinin dışında lojistik regresyon yöntemiyle daha doğru ve daha tutarlı yordama yapılabilmektedir. Girdi değişkenlerinin modeldeki katkılarına göre ağırlıklandırıldığı bu yaklaşımda tek bir regresyon denklemi yerine farklı özellikler için çok sayıda lojistik denklem elde edilmektedir (Perlich, Provost ve Simonoff, 2002).

Karar Ağaçları

Bir karar ağacı oluşturma problemi kendi kendini tekrarlar şekilde ifade edilmektedir. Öncelikle kök düğüme (herhangi bir ata düğümü olmayan ve dolayısıyla en üstte olan düğüm) yerleşecek şekilde bir özellik belirlenir ve olası her bir değer için bir şube (dal) oluşturulur (Fayyad ve Irani, 1992). Bu işlem örnek kümeyi her bir özellik değeri için alt kümelere böler. Süreç sadece her bir dala ulaşan örnekleri kullanarak her bir dal için ardışık olarak tekrar edilir. Eğer herhangi bir durumda bir düğümdeki tüm örnekler aynı sınıfa dahil olurlarsa ağacın o parçasının (düğümü) geliştirilmesi durdurulur. Çünkü artık farklı sınıflara ayrışma olmayacaktır (Quinlan, 1993). Karar vermek için geriye kalan tek şey, farklı sınıflarda bir dizi örnek verildiğinde, hangi özelliğin nasıl bölüneceğini belirlemektedir. Aşağıda hava durumuyla ilgili olarak genel görünüş ve sıcaklık özelliği için oyun oynama durumuna ilişkin sonuçlar yer almaktadır. Her bölünme için olası 2 ihtimal (Evet-Hayır) bulunmaktadır ve Şekil 1'de gösterildiği gibi en üstte özellik olacak şekilde sınıflara ayrılmaktadır.



Şekil 1. Hava Durumu Verisi İçin Ağaç Kütükleri (Witten ve Frank, 2005)

Şekilde verilen ağaç yapısına ilişkin dallanmalardan hangisinin en iyi seçim olduğunu dikdörtgen şeklinde gösterilen yapraktaki örneklerin hangi sınıfa dahil olduğuna bakarak karar verebiliriz. Yapraklarda evet ve hayır sınıflarının sayıları gösterilmektedir. Sadece Evet veya Hayır şeklinde yaprakta tek bir sınıf oluştuğunda tekrardan ayrılmaya gerek kalmayacak ve aşağıya doğru yinelenen dallanma süreci sona erecektir.

Bu işlemin mümkün olduğunca kısa bir sürede gerçekleşmesini istediğimiz için küçük ağaçları inceleriz. Eğer her bir düğümün saflığını (purity) ölçseydik bu durumda en saf çocuk düğümleri üreten özellikleri seçmemiz gerekecektir. Şimdi biraz zaman ayırın ve hangi özelliğin en iyisi olacağını düşünün.

Kullanacağımız saflığın ölçüsü bilgi (information) olarak adlandırılır ve bit olarak belirlenen birimlerle ölçülür. Bit, ağacın bir düğümü ile ilişkili olarak yeni bir Evet (oyun oynanacak) veya Hayır (oyun oynanmayacak) şeklinde sınıflandırılmada gereken bilgi miktarını temsil etmektedir. Elbette, verilen bu sayılarda özel bir şey yoktur ve gerçek değerlerinden bağımsız olarak bunlar arasında benzer bir ilişki bulunmaktadır. Böylece bir listeye bir başka ölçüt ekleyebiliriz. Elde edilen bilgi daha önce gösterilen çok aşamalı özelliğe uygun olmalıdır. Dikkat çekici bir şekilde, tüm bu özellikleri karşılayan tek bir fonksiyon olduğu görülmektedir ve bu fonksiyon bilgi miktarı veya entropi olarak isimlendirilir (Rokach ve Maimon, 2008). Aşağıdaki eşitlikte bilgi miktarının matematiksel olarak nasıl elde edildiği gösterilmektedir.

Entropy (p_1, p_2, …, p_n) = - p_1.logp_1 – p_2.logp_2, … , - p_n.logp_n

Buradaki (-) eksi işaretlerinin sebebi p_1,p_2,…,p_n kesirlerinden kaynaklanma logaritma fonksiyonunun kuralı gereği A/B şeklinde kesirli ifadelerin logaritması alınırken taban aynı olmak üzere paydaki ifadenin önüne eksi işareti konulmaktadır ($\log_a {}^{A}/_{B} = \log_a A - \log_a B$). Her ne kadar kesirlerin önünde eksi işareti olsa da elde edilen bilgi miktarı gerçekte pozitif olacaktır.

Genellikle logaritmalar 2 tabanında verilir (log2 x) ve bu durumda entropi olarak adlandırılan bilgi miktarları bit olarak tanımlanmaktadır. Bu bitler bilgisayardaki kullanılan bitlerin (1-0) genel bir türüdür.

Entropide yer alan p1,p2,…,pn ifadeleri toplamı 1 olan kesirler anlamına gelmektedir. Örneğin;

info ([2,3,4]) = entropi (2/9, 3/9, 4/9)

Yukarıdaki formülde görüldüğü gibi p_1,p_2,p_3 olarak belirtilen 2/9, 3/9 ve 4/9 kesirlerinin toplamı 1'e eşit olacaktır. Dolayısıyla çok aşamalı özelliklere ilişkin kararlar aşağıda verilen genel formül yardımıyla elde edilebilir.

entropi (p,q,r) = entropi (p,q+r) + (q+r). entropi ($\frac{q}{q+r}$, $\frac{r}{q+r}$)

Verilen formülde p+q+r = 1 olduğu unutulmamalıdır.

info ([2,3,4]) = $-\frac{2}{9}$ x log $\left(\frac{2}{9}\right)$ - $\frac{3}{9}$ x log $\left(\frac{3}{9}\right)$ - $\frac{4}{9}$ x log $\left(\frac{4}{9}\right)$ = $-\frac{2}{9}$ x (log2 – log9) - $\frac{3}{9}$ x (log3 – log9) - $\frac{4}{9}$ x (log4 – log9)

$$= \frac{2\log 2+2\log 9}{9} - \frac{3\log 3+3\log 9}{9} - \frac{4\log 4+4\log 9}{9}$$

$$= \frac{-2\log 2-3\log 3-4\log 4-9\log 9}{9}$$

Yukarıda verilen formül bilgi miktarının pratikte nasıl hesaplandığını gösteren bir formüldür.

Güvenirlik: Ne Öğrendiğimizin Değerlendirilmesi

Değerlendirme veri madenciliğinde gerçek anlamda bir ilerleme sağlamak için anahtar rolündedir. Eldeki veri dosyasından örüntü çıkarmak ve bu örüntülerden çıkarımda bulunmanın birçok farklı yolu vardır. Fakat belirli bir problem durumunda hangi yönteminin kullanılacağını belirleme aşamasında farklı yöntemlerin nasıl çalıştığını değerlendirmek ve birbirleriyle karşılaştırmak için sistematik yollara ihtiyacımız bulunmaktadır. Değerlendirme işlemi ise ilk görünüşte göründüğü kadar basit bir işlem değildir.

İki farklı yönteminin hangisinin daha iyi çalıştığını görebilmeniz için eğitim ve test setine ihtiyaç duyulmaktadır. Ama aslında eğitim setindeki performansın nasıl olduğunu öğrenmek bağımsız bir test setindeki performansın iyi bir göstergesi olmayacaktır. Elde edilen veri setleri üzerinde yapılacak deneysel çalışmalara dayalı olarak performans sınırlarını tahmin etme yollarına ihtiyacımız olacaktır (Breiman ve Friedman, 1984). Daha açık bir ifadeyle aynı veri seti üzerinden öğrenme yöntemini eğitmek ve sonrasında yine aynı veri seti üzerinden bu yöntemi test etmek gerçeği yansıtan bir yaklaşım olmayacaktır. Bu sebeple eğitim setindeki hata oranı gerçekteki performansın iyi bir göstergesi olmayabilir. Çünkü sınıflandırıcı aynı eğitim verilerinden öğrendiği için bu verilere dayalı olarak yapılacak herhangi bir hesaplama işlemi de iyimser olacaktır (Witten ve Frank, 2005).

Veri madenciliğinde ilgilendiğimiz şey eski verilere dayalı geçmiş performans yerine yeni verilere dayalı olarak gelecekteki muhtemel performans olacaktır. Zaten hali hazırda eğitim kümesindeki her bir örneğin hangi sınıfta olduğu bilinmektedir. Genel olarak biz bu sınıflandırma ile ilgilenmiyoruz ancak amacımız tahminden ziyade veri temizlemek ise sınıflamalara da dikkat edilmektedir. Yeni veri üzerinden bir sınıflayıcının performansını tahmin etmek için sınıflandırıcının elde edildiği veri setinden daha farklı bir veri seti üzerinden hata oranını değerlendirmemiz gerekmektedir. Çalışmalarda hem eğitim verilerinin hem de test verilerinin araştırdığımız problemin temsili örnekleri olduğunu varsaymaktayız (Sumathi ve Sivanandam, 2006).

Bazı durumlarda test verileri doğadaki eğitim verilerinden farklı olabilmektedir. Örneğin kredi risk problemine ilişkin örneklerimiz bulunsun. Bu örnekler üzerinden bankanın New York ve Florida'daki şubelerden eğitim verileri aldığını ve bu eğitim verisi üzerinden elde edilecek sınıflandırıcının Meksika'daki yeni bir şubede ne kadar iyi performans göstereceğini öğrenmek istediğini varsayalım. Muhtemelen New York sınıflandırıcısını değerlendirmek için Florida verilerini test verileri olarak kullanmak ve Florida sınıflandırıcısını değerlendirmek için New York verilerini kullanmak uygun olacaktır. Eğer veri setleri veriyi eğitme işleminde önce bir araya getirilirse test verilerindeki performans muhtemelen gelecekte tamamı ile farklı bir eyaletten elde edilecek verilerin performansının iyi bir göstergesi olmayacaktır (Witten, Frank ve Hall, 2016). Eğitim alanında yapılan çalışmalarda farklı bir öğrenme yönteminin öğrencilerin okulda gördükleri dersler üzerinde etkili olup olmadığı araştırılmıştır. Lopez ve diğerleri (2012) tarafından yapılan çalışmada

öğrencilerin Moodle öğrenme yönetim sistemindeki forum kullanım verilerinin ders başarısının önemli bir göstergesi olup olmadığı araştırılmıştır. Çalışmada ayrıca sınıf değişkeninin (ders başarısı) bilinmediği durumlarda kümeleme algoritmaları ile aynı sonuca ulaşılıp ulaşılmayacağı test edilmiştir.

Çoğu durumda eğitim verileri elle sınıflandırılmalıdır ve tabiî ki hata oranını elde etmek için test verileri de benzer şekilde sınıflandırılmaktadır. Bu durum eğitim, doğrulama ve test için kullanılabilecek veri miktarını sınırlandırır ve bu sınırlı veri ile en iyi performans nasıl elde edileceği bir soru haline gelir. Bu veri kümesinden test için belirli bir miktar (%20 - %30) tutulur ve bu işleme bekletme prosedürü denilir ve sonrasında geriye kalan miktar eğitim için kullanılır. Ancak eğer gerekli ise eğitim için kullanılacak verinin bir kısmı doğrulama verisi olarak ayrılabilir (Mitchell, 1997).

Çapraz Geçerlik

Elinizde eğitim ve test için sınırlı miktarda veri olduğunda ne yapacaksınız ? Bekletme yönteminde eldeki verinin bir miktarını test etmek için ayırırken geri kalan kısmı eğitim için kullanılmaktadır. Hatta eğitim için ayrılan verilen bir kısmı doğrulama için de kullanılabilmektedir. Pratik olarak eldeki verinin üçte birini test için geri kalan üçte ikilik kısmı ise eğitim için kullanmak oldukça yaygın bir yöntemdir (%66=Eğitim, %34=Test verisi).

Elbette eğitim (ya da test) için kullandığınız örneklem grubu evrenin iyi bir temsilcisi olmayabilir. Genelde bir örneklemin iyi bir temsil olup olmadığını doğrudan anlayamazsınız. Ancak sizin için önemli olabilecek çok basit bir kontrol vardır. Bu yaklaşımda verilerin tamamının yer aldığı çalışma evreninizde yer alan her bir sınıfın eğitim ve test verilerinde doğru bir oranda yer alması gerekmektedir. Ancak bu sayede örneklemin evreni temsil gücü artacaktır. Ancak kötü bir şansınız varsa ve aldığınız örneklemdeki bir sınıf için tüm örneklerde kayıp veri varsa ve siz bu verilerden elde edilecek sınıflandırıcının test verileri üzerinde iyi bir performans göstermesini bekleyemezsiniz. Böyle bir durumda bu verilerin hiçbiri eğitim kümesinde yer almadığı için test kümesindeki verilerin aşırı temsil edilmesi sebebiyle sonuçlar daha da kötüleşecektir. Bunun yerine evrende yer alan her bir sınıfın eğitim ve test verilerinde düzgün bir şekilde temsil edildiğini garanti edebilmek amacıyla tesadüfü örnekleme yapıldığından emin olmanız gerekmektedir. Bu yönteme tabakalandırma veya tabakalı bekletme yaklaşımı adı verilmektedir. Her ne kadar bu yöntemi uygulamak genellikle kayda değer olsa da bu sadece eğitim ve test verilerindeki eşit olmayan veya dengesiz bir temsili ortadan kaldırmak için ilkel bir koruma olarak görülmektedir (Witten, Frank ve Hall, 2016).

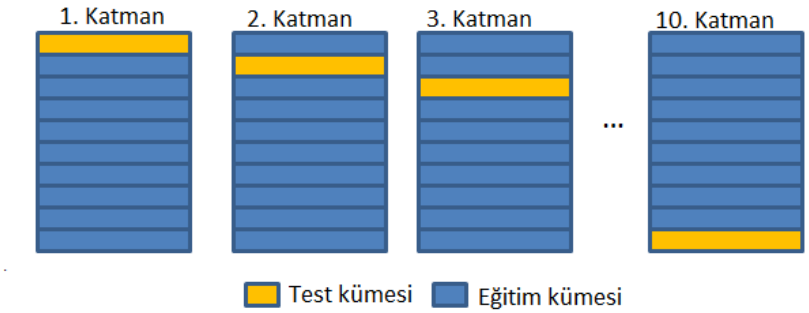
Belirli bir örneklem seçiminden kaynaklanan yanlılığı azaltmanın daha genel bir yolu farklı örneklem grupları üzerinden eğitim ve test verileri için tüm süreci birkaç kez tekrar etmektedir. Her bir tekrarlama işlemini verinin belli bir miktarı, örneğin eğitim için üçte ikisi, eğer mümkünse tabakalı ve rastgele seçilmiş bir şekilde eğitim için kullanılırken geri kalan miktarı test için kullanılabilir. Farklı tekrarlardan elde edilen hata oranları ortalaması alınarak daha genel bir hata oranı elde edilebilir. Buna hata oranını hesaplamak için tekrarlı bekletme yöntemi adı verilmektedir.

Sadece bir bekletme yönteminin uygulandığı durumlarda, eğitim ve test verilerinin rollerini değiş-tokuş ettiğini düşünebilirsiniz. Yani test verileri üzerinden sistemi eğitirken, eğitim verileri üzerinden bu sistemi test edebilirsiniz. Böylece elde ettiğiniz iki sonucun da ortalamasını alarak eğitim ve test verilerindeki eşit olmayan temsil sorununu azaltmış olursunuz.

Ne yazık ki bu sadece eğitim ve test verileri için %50 - %50 oranında bir bölünme durumunda mantıklı olmaktadır. Ancak bu yaklaşım genellikle uygun bir bölünme olarak görülmemektedir. Bunun yerine eldeki verilerin yarıdan daha fazlasını eğitim için kullanmak daha iyi bir yaklaşımdır. Bununla birlikte **çapraz geçerleme** olarak isimlendirilen önemli bir istatistiksel teknik bulunmaktadır. Çapraz geçerleme sabit bir katlanma ya da verilerin bölünme sayısına karar veriyoruz örneğin kullandığımız sabit sayı 3 olsun. Daha sonra veriler yaklaşık olarak eşit üç parçaya ayrılır ve her biri sırasıyla test için kullanılırken geri kalanları eğitim için kullanılmaktadır. Yani böylece eldeki verinin test edilmesi için üçte biri kullanırken eğitim için eldeki verilerin üçte ikisi

kullanılmış olmaktadır. Bu şekilde işlemler tekrar edildiğinde her bir örnek test için bir kez kullanılmış olacaktır. Buna üçlü çapraz geçerlik denir ve eğer tabakalandırma işlemi yapılmışsa buna tabakalı üçlü çapraz geçerlik denilmektedir (Kohavi, 1995).

Elinizdeki tek ve sabit bir veri olduğunda öğrenme tekniğinin hata oranını tahmin edebilmek için kullanılan standart yöntemde 10 katı çapraz geçerleme yaklaşımı uygulanmaktadır. Bu yaklaşımda verilerin tamamının yer aldığı çalışma evreni rastgele 10 sınıfa ayrılmaktadır ve her bir sınıfın tam veri kümesinde yaklaşık olarak aynı oranda temsil edilmesi gerekmektedir. 10 katlı çapraz geçerleme sürecinin nasıl gerçekleştiği daha kolay anlaşılması amacıyla Şekil 2 üzerinde gösterilmiştir.



Şekil 2. On Katlı Çapraz Geçerleme

Bu süreçte her bir sınıf sırasıyla test için ayrı bir kenarda bekletilirken geriye kalan dokuz sınıf üzerinden öğrenme süreci gerçekleşir ve bir kenarda bekletilen 1/10'luk veriler üzerinden hata oranı hesaplanmaktadır. Böylece öğrenme yöntemi farklı eğitim verileri üzerinden (her birinin birçok ortak noktası bulunan) toplam 10 defa tekrar edilerek gerçekleşmektedir. Son olarak 10 farklı hata teriminin ortalaması alınarak genel bir hata terimi elde edilmektedir (Refaeilzadeh, Tang ve Liu, 2007).

Bootstrap Yöntemi

Bu yöntem tekrarlı örnekleme olarak tanımlanabilecek istatistiksel işlemlere dayanmaktadır. Öncesinde eğitim veya test için eldeki veri setinden bir örneklem alınır ve bu örneğin üzeri çizilir. Yani aynı örnek bir kez seçildiğinde tekrardan seçilmemektedir. Bu yaklaşım futbol oyunu için takımlar oluşturmaya benzer. Nasıl ki bir oyuncuyu farklı iki takıma seçemeyeceğiniz gibi Bootstrap yönteminde de aynı örnek iki kez seçilemez. Ancak veri kümesindeki örnekler insanlar gibi değildirler. Çoğu öğrenme yöntemi aynı örneği iki defa kullanmaktadır ve eğer bu eğitim kümesinde iki defa yer almışsa sonuçlarda farklılık oluşacaktır. Matematiksel olarak konuşmak gerekirse aynı nesnenin birden fazla görüntüsü olması durumunda "küme" olarak tanımlanan gruplardan bahsetmek anlamlı olmayacaktır (Ibarguren et al., 2014).

Bootstrap yönteminin mantığı bir eğitim kümesi oluşturabilmek için elimizdeki veri setini yer değiştirerek örnekleyebilmektir. Bu aşamada gizemli bir şekilde 0,632 Bootstrap sayısının ne olduğu ve nasıl elde edildiği gösterilecektir. Bunun için n örnekten oluşan bir veri seti yer değiştirilerek n örnekten oluşan başka örneklem elde edilecek şekilde örneklenmektedir. Bu ikinci veri setinde yer alan bazı örnekler tekrar edeceği için test örnekleri olarak kullanılacak ve orijinal veri kümesinde başka yerde kullanılmış olan örneklerin olması gerekmektedir.

Peki belirli bir örneğin eğitim kümesinde yer almama ihtimali nedir? Her bir seferde verilen örneğin eğitim kümesinde yer alma olasılığı $\frac{1}{n}$ iken bu kümede yer almama olasılığı $(1 - \frac{1}{n})$ olarak hesaplanmaktadır. Bu olasılıkların verilen kümede yer alma olasılıklarının çarpımı aşağıda verilen eşitlik yardımıyla hesaplanmaktadır.

$$(1-\frac{1}{n})n \approx e^{-1} = 0{,}368$$

Yukarıda verilen eşitlikte e ile gösterilen sembol doğal logaritmanın tabanıdır ve 2,7168 değerine eşittir. Ancak bu gösterimin hata oranına eşit olmadığını unutmamanız gerekmektedir. Bu eşitlik belirli bir örneğin herhangi bir şekilde seçilmeme olasılığını göstermektedir. Böylece oldukça büyük bir veri kümesi için test kümesi veri setinin yaklaşık %36,80'ini kapsayacaktır ve dolayısıyla eğitim kümesi elinizdeki veri setinin %63,20'sini kapsayacaktır. Daha önce gizemli bir sayı olarak tanımlamış olduğumuz 0,632 Bootstrap sayısının nereden geldiğini öğrenmiş bulunmaktayız. Bazı örnekler eğitim kümesinde tekrar kullanılarak orijinal veri setindeki n toplam sayıya ulaşılmaktadır (Kushary, 2015).

Eğitim kümesi üzerinden bir öğrenme sistemini eğiterek elde edilen bu sayısal değer ve test kümesi üzerinden elde edilen hata değeri gerçek hatanın kötümser bir tahmini olacaktır. Çünkü eğitim seti, boyutu n olmasına rağmen yine de örneklerin %63'ünü kapsamaktadır ve 10 katlı çapraz geçerlemede verilerin %90'ının kullandığı düşünüldüğünde bunun daha büyük bir oran olmadığı görülmektedir. Bunu telafi etmek için eğitim kümesi içindeki örnekler üzerinden yeniden yerleştirme işlemi sonucunda elde edilen hata ile test kümesi üzerinden elde dilen hata oranı ile birleştirme yapılır. Yeniden yerleştirme sonucunda elde edilen hata değeri gerçek hata değerinin oldukça iyimser bir tahmini olacağı için tek başına hata terimi olarak kullanılmaması gerekmektedir. Ancak Bootstrap yöntemi hata değerini eğitim ve test verileri üzerinden elde edilen hataların birleşiminde oluşturur ve aşağıda gösterilen nihai bir hata değeri olarak hesaplanmaktadır (Bradley ve Tibshirani, 1993).

$$e = 0{,}632 \times e_{\text{test verileri}} + 0{,}368 \times e_{\text{eğitim verileri}}$$

Ardından tüm Bootstrap yöntemi eğitim kümesi için farklı örneklemlerin yer değiştirilmesi suretiyle birkaç kez tekrarlanmaktadır ve elde edilen sonuçların ortalaması alınmaktadır.

Bootstrap yöntemi küçük veri setleri için hata oranını tahmin etmesi bakımından en iyi yöntem olarak kabul edilebilir. Buna rağmen tıpkı bir tanesini kapsam dışı bırakma yönteminde olduğu gibi tamamen yapay ve özel bir durum üzerinden sahip olduğu dezavantajın ne olduğu gösterilebilir. Aslında daha önce düşündüğümüz veri kümesi tamamı ile tesadüfi olarak iki sınıfa ayrılmış olsun. Herhangi bir tahminde bulunma kuralı için gerçek hata oranı %50 olsun. Ancak eğitim kümesini düşünelim ve bir taslak öğrenme yönteminin yeniden yerleştirme sonucunda hatası (e= 0) sıfır olan %100 başarılı bir sonuç verecektir. Bu durumda 0,632 Bootstrap yöntemi bunu 0,368 ile ağırlıklandıracak ve sonuç olarak 0,632 × %50 + 0,368 × %0 olmak üzere genel hata terimi %31,60 olarak hesaplanacaktır ki bu değer yanıltıcı şekilde iyimser olacaktır.

Kestirimin Gücü: Hatanın büyüklüğünü dikkate alma

Sınıflandırmanın doğruluğuna göre yapılan değerlendirme işlemi üstü kapalı bir şekilde farklı hatalarının aynı anlama gelmediğini varsaymaktadır. Bu duruma ilişkin bir örnek kredi borçlarında görülebilir. Borcunun ödemeyen birine kredi vererek yapılan hata daha önce hiç kredi çekmemiş birine kredi vermeyerek yapılan hatadan oldukça büyüktür.

Evet-hayır, ödünç para verme-vermeme, kredi verme -vermeme, şüpheli bir parçayı atık veya değil, vb. şekillerde iki sınıfı olan durumlarda bu duruma ilişkin sonuçlar Tablo 1'de gösterildiği gibi olası 4 farklı sonuç bulunmaktadır. Doğru pozitif (DP) ve doğru negatif (DN) doğru sınıflama sonuçlarıdır. DP gerçek durum pozitifken test sonucunda pozitif çıkan durumlardır. DN gerçek durum negatifken test sonucunda negatif çıkan durumlardır. Yanlış pozitif (YP) sonuç aslında negatif (hayır) iken yanlış bir şekilde çıktının pozitif (EVET) olarak tahmin edilmesidir. Yanlış negatif (YN) sonuç aslında pozitif iken yanlışlıkla negatif olarak tahmin edilmesidir. Doğru pozitif oranı (DPO) gerçek durumu pozitif olanlar tüm durumlara bölünerek elde edilirken [DP/(DP+YN)] yanlış pozitif oranı gerçek durumu negatif olan tüm durumlara bölünerek [YP/(YP+DN)] elde edilmektedir (Powers, 2011).

Tablo 1. İki Sınıflı Bir Tahmine İlişkin Farklı Sonuçlar

		GERÇEK DURUM		
		POZİTİF	NEGATİF	TOPLAM
TEST SONUCU	POZİTİF	DOĞRU POZİTİF (DP)	YANLIŞ POZİTİF (YP)	DP+YP
	NEGATİF	YANLIŞ NEGATİF (YN)	DOĞRU NEGATİF (DN)	YN+DN
	TOPLAM	DP+YN	YP+DN	DP+YP+YN+DN

Sonuçlara ilişkin oranların ardından genel başarı oranı doğru sınıflamaların sayısının toplam sınıflama sayısına bölümüdür.

$$\text{Genel Başarı} = \frac{DP+DN}{DP+DN+YP+YN}$$

Bu sınıflama için hata oranı ise 1'den genel başarı oranının çıkarılması sonucunda elde edilmektedir. Duyarlılık ve seçicilik kavramlarının da bilinmesinde yarar vardır. Duyarlılık: Testin gerçek pozitif durumlar içinden pozitif olan durumları ayırma yeteneğidir. Aynı zamanda doğru pozitif oranı duyarlılık olarak tanımlanmaktadır.

$$\text{Duyarlılık (Sensitivity)} = \frac{DP}{DP+YN}$$

Seçicilik: Testin gerçek negatif durumlar içinden negatif olan durumları ayırma yeteneğidir.

$$\text{Seçicilik (Specifity)} = \frac{DN}{DN+YP}$$

Yukarıda anlatılan duyarlılık ve seçicilik değerleri, testin araştırılan durumla ilgili olan ve ilgili olmayanları birbirinden ne kadar iyi ayırt edip etmediğini tanımlamaktadır (Johari, 2016). Yanlış pozitif oranı I. Tip Hata ve Yanlış negatif oranı II. Tip hata olarak kabul edilmektedir.

Çok sınıflı bir tahminde, testten elde edilen sonuçlar hata matrisi adı verilen iki boyutlu her bir sınıf için satır ve sütunları olan bir tablo üzerinde gösterilmektedir. Matrisin her bir elemanı sütunu gerçek değeri, satırı ise tahmin edilen (test sonucu) örnek sayısını gösterecek şekilde tanımlanmaktadır (bazı durumlarda satırlar gerçek durumu, sütunlar test sonuçlarını göstermektedir). İyi sonuçlar esas köşegen üzerindeki (sol üst köşeden sağ alt köşeye çizilen doğru parçası) sayıların oldukça büyük ve köşegen dışındaki elemanların küçük hatta ideal olanı sıfır olması durumunda elde edilir.

ROC (Receiver Operating Characteristic) eğrileri olarak bilinen ve yüksek pozitiflik oranına sahip test örneklerini seçmeye çalışan eğriler bir sınıflandırıcının performansını hatanın türüne ve sınıf dağılımına bakmaksızın betimlemeye çalışmaktadır. Bu grafiklerde yatay eksende (x-ekseni) örneklerde yer alan negatif sayısına karşılık gelen pozitif örnek sayıları yüzde cinsinden gösterilirken dikey eksende örneklerdeki pozitif sayısına karşılık gelen negatif örnek sayıları yüzde cinsinden gösterilmektedir (Sinha ve May, 2005). Kappa istatistiği ya da Kappa değeri beklenen ve gözlenen değerlerin karşılaştırıldığı sayısal bir değerdir. Bunun yanında tesadüfi şans faktörünü de hesaba kattığından daha az yanıltıcı bir değerdir. Beklenen ve gözlenen doğruluk hesabı Kappa istatistiğinde aynı anda kullanılmakta ve karışıklık matrisi üzerinden kolaylıkla belirlenebilmektedir. Bu yöntemde satırdaki değerler gerçek değerleri, sütundaki değerler ise tahmin edilen değerleri göstermek üzere gözlenen doğruluk oranından beklenen doğruluk oranı çıkartılıp elde edilen bu değer 1-bekelenen doğruluk oranına bölünmektedir. Kappa istatistiğinin yorumlanması konusunda kesin standart bir yorum olmasa da genel olarak 0.00-0.20 arası düşük, 0.21-0.40 arası kayda değer, 0.41-0.60 arası orta düzeyde; 0.61-0.80 arası önemli ve 0.81-1.00 arası mükemmel olarak tanımlanmaktadır (Landis ve Koch, 1977). Veri madenciliğinde sayısal tahminin değerlendirilmesinde farklı hata ölçütleri bulunmaktadır. Ortalama Karesel hata, hataların ortalama karekökü, ortalama mutlak hata, kareli bağıl hata, bağıl hataların karekökü ve göreceli bağıl hata sayısal tahminin ne kadar doğru olduğunu belirlemede kullanılan hata ölçütleridir (Witten ve Frank, 2005). Bağıl hata değerleri çıktı değişkenin temel tahmin edilebilirliğini veya tahmin edilemezliğini dengelemeye çalışırken, kareli hata ölçütleri ile hataların karekökü ölçütleri hataların aynı boyuta indirgenmesi sağlanmaktadır. Veri madenciliğinde kullanılan F-ölçütü ise kesinlik ve geri çağırma değerlerinden elde edilmektedir. Özellikle tıp ve medikal alanında büyük öneme sahip olan kesinlik (*precision*) doğru (pozitif) olarak

tahmin edilen bir durumdaki başarıyı göstermektedir. Geri getirme (*recall*) Pazarlama ve market araştırmasında daha büyük öneme sahiptir ve pozitif durumların ne kadar başarılı tahmin edildiğini göstermektedir (Schwenke ve Schering, 2007).

PROGRAMIN TANITILMASI, ANALİZLER, KESTİRİM VE SINIFLAMA ÖRNEKLERİ

WEKA programının ana arayüzü olan keşfedici (explorer) menü seçimi ve form doldurma imkanı sunarak tüm işlemlere kolaylıkla ulaşım imkanı sağlamaktadır. Şekil 3'te gösteriliği gibi WEKA ana ekranındaki pencere tıklanarak altı farklı sekme altında WEKA programının desteklendiği farklı madencilik görevleri listelenmektedir. Knowledge Flow, WEKA'nın içindeki bütün özelliklerin bir ekran üzerinden birden fazla kullanımına izin veren bir özelliğidir. Explorer ekranında sadece tek bir işlem yapılabilirken, Knowledge Flow işlerin arka arkaya farklı işlemler üzerinden tekrar tekrar çalışmasını sağlayan bir özelliktir. Explorer ekranı altında yapılan işlemlerde her şey otomatik ve hazır bulunur. Knowledge Flow'da ise bu işlemler kullanıcı tarafından oluşturulması gerekmektedir (Şeker, 2016).

Şekil 3. WEKA Ana Ekranı

Düşünün ki elinizde bir takım veriler var ve siz bu verilerden karar ağacı elde etmek istiyorsunuz. İlk olarak verinizi hazırlamanız daha sonra keşfediciyi çalıştırmanız ve verilerini programa yüklemeniz gerekmektedir. Sonrasında bir karar ağacı oluşturma yöntemi seçip, bir ağaç oluşturun ve elde ettiğiniz çıktıyı yorumlayın. Bu işlem farklı bir karar ağacı algoritması ve farklı bir değerlendirme yöntemi ile tekrarlamak oldukça kolaydır. Keşfedici menüsü altında elde ettiğiniz sonuçlar arasında ileri-geri geçişler yapabilir, farklı veri setleri üzerine inşa edilmiş modelleri değerlendirebilir ve modellerin yapmış olduğu sınıflandırma hataları da dahil olmak üzere hem modelleri hem de veri setlerinin kendisini grafiksel olarak görselleştirebilirsiniz.

Aşağıda Şekil 4 ve Şekil 5'te keşfedici penceresinin üst kısmında yer alan altı farklı sekme kısaca tanımlanmıştır. Bu sekmelerin her biri elinizdeki verilerle gerçekleştirebileceğiniz farklı işlemleri göstermektedir.

1. PREPROCESS: (ön hazırlık): veri setini seçmenizi ve bunu farklı yöntemlerle düzenlemenizi sağlar
2. CLASSIFY: (sınıflama): sınıflama veya tahminde bulunacak öğrenme yöntemini eğitir ve bunları değerlendirir
3. CLUSTER: (kümeleme): veri kümeleri için kümeleri öğrenin
4. ASSOCIATE: (birliktelik): veri setiniz için birliktelik kurallarını öğrenir ve bunları değerlendirir

5. SELECT ATTRIBUTES: (özellik seçme): veri kümenizdeki birbirleriyle alakalı özellikleri seçer
6. VISUALIZE: (görselleştirme): veri kümenizin iki boyutlu farklı grafiklerini görmenizi ve bunlar arasındaki etkileşimi belirleyebilmenizi sağlamaktadır.

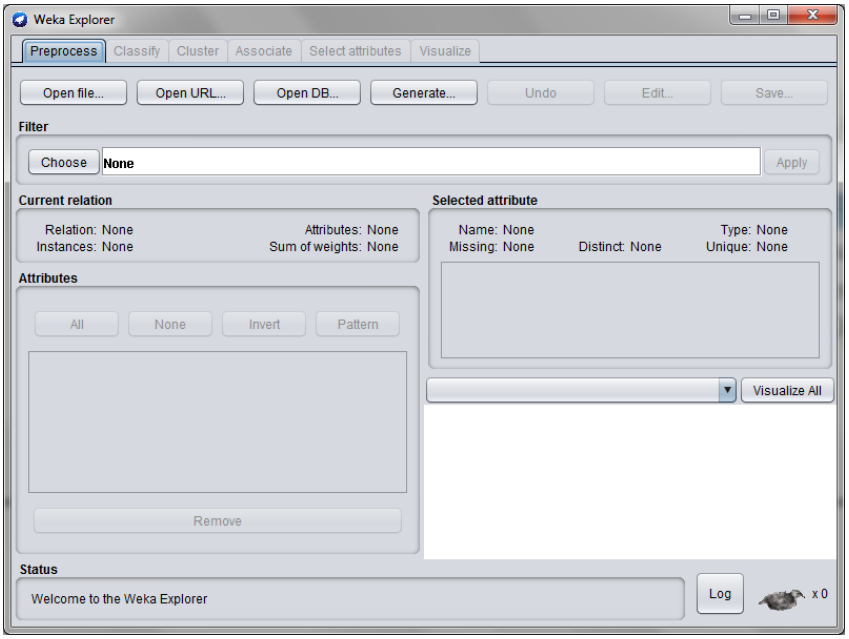
Her bir seçenek bir dizi olanağa erişim imkanı sağlamaktadır. Şuana kadar sadece ön hazırlık ve sınıflama seçenekleri yüzeysel olarak ele alınmıştır. Daha ileri düzeyde analiz yapmak isteyen araştırmacıların kümeleme (*cluster*), birliktelik kuralları (*associate*), özellik seçme (*select attributes*) ve görselleştirme (*visualize*) seçeneklerini detaylı bir şekilde incelemeleri önerilmektedir.

Veriyi Analiz İçin Hazırlama

Veriler genellikle tablo veya veri tabanı şeklinde sunulmaktadır. Buna rağmen, WEKA programının verileri depolama yöntemi ARFF formatında elimizdeki tabloya girmiş olduğunuz verileri kolaylıkla ARFF dosyası veri dosyasındaki örneklerin toplu listesini içerir ve her örnek için belirlenen öznitelik değerleri virgülle ayrılarak gösterilmektedir. Çoğu tablo ve veri tabanı programı veri dosyanızı virgülle ayrılmış değerler (comma-seperated value=csv) türünde çevirmenize imkan vermektedir. Bu dosya türünde veri dosyasındaki değerler arasında virgül konularak depolama yapılmaktadır. Bunu yaptıktan sonra dosyanızı bir metin belgesine veya programına yüklemeniz yeterli olacaktır. Sonrasında veri dosyanızın adına @relation eklentisi ile ilişkileri, @attribute eklentisi ile öznitelik bilgilerini ve @data eklentisi ile verinizi ham metin belgesi olarak kaydediniz. Örneğin daha sonra bahsedilecek olan PISA verileri için bir Excel veri dosyasını .csv uzantılı olarak farklı kaydedilir ve bu dosya program üzerinden açılarak .arff uzantılı WEKA veri dosyasına dönüştürülür. Buna rağmen, aslında ARFF dosyasını kendiniz oluşturmanız için bu adımları izlemenize gerek yoktur, çünkü keşfedici CSV uzantılı dosyaları doğrudan okuyabilmektedir.

Veriyi Programa Tanıtma

Şimdi sırada elinizdeki verileri keşfediciye yükleyip bunu analiz etmeye çalışalım. Şekil 4'te gösterilen ekrana ulaşabilmek için WEKA programını başlatınız. Daha sonra uygulamalar başlığı altında yer alan beş farklı arayüzden biri olan keşfediciyi seçiniz.

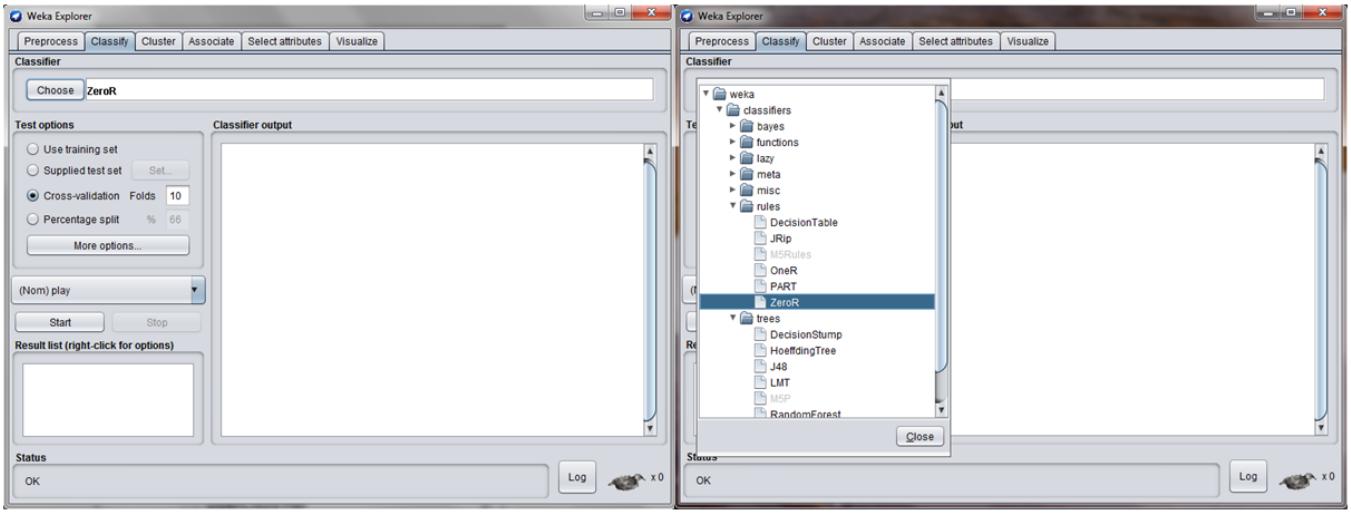


Şekil 4. Veri Hazırlama Ekranı

Daha sonra *Dosya Aç (Open File)* düğmesi tıklanarak daha önceki dosyalarınızdan bir tanesi tekrardan görebilirsiniz. Bilgisayarınızda kayıtlı olan dosyanızı tanımlayarak programa verilerinizi göstermiş olacaksınız. Eğer verileriniz WEKA programının temel veri dosya uzantısı olan ARFF formatında değilse dosya türünü CSV olarak seçmeniz gerekecektir. Programda *.csv* uzantılı bir dosya belirlediğinizde bu dosyanız otomatik olarak ARFF uzantılı bir dosya türüne dönüştürülecektir. Dosyayı yükledikten sonra karşınıza çıkan ekranda size elinizdeki veri kümesi hakkında bilgi verilmektedir.

Analizlerin Gerçekleştirilmesi: Karar Ağacı Oluşturma

Veri madenciliğinde en fazla kullanılan algoritmalardan biri olan C4.5 karar ağacı öğrenme yöntemi J.48 algoritması ile çalışmaktadır ve bu algoritma WEKA programı tarafından C5.0 uygulaması piyasaya sürülmeden önceki herkes tarafından kullanılabilen bir versiyonudur (Kaur ve Chhabra, 2014). Şekil 5(a)'da gösterildiği gibi sınıfla (*CLASSIFY*) düğmesini tıkladığınızda açılan pencerede seçme (*CHOOSE*) düğmesini tıkladığınız takdirde Şekil 5(b)'de gösterilen ekran karşınıza gelecektir. Ekranda analiz işlemi gerçekleştirmediği için sağ alt köşede yer alan çıktı penceresinde her hangi bir sonuç bulunmamaktadır. Sınıflama için kullanılacak algoritma ve test türünün belirlenmesinin ardından tek yapmanız gereken Başla (*START*) düğmesine tıklamanızdır.

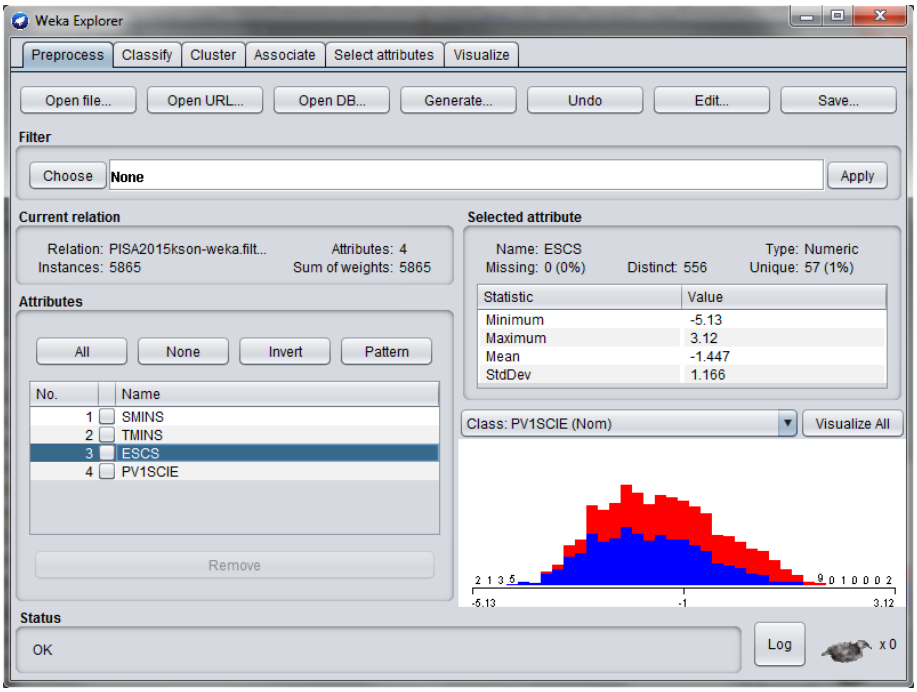


Şekil 5. Başlatma Ekranı: (a) Seç Penceresi ve (b) Sınıflama Penceresi

Analiz işlemini gerçekleştirebilmek için önce sol üst tarafta yer alan seç düğmesini tıklayarak hiyerarşik olarak listelenmiş olan menüden Ağaçlar (*Trees*) bölümünü tıklayıp buradan J.48 yöntemini seçiniz. Şuan için tek yapmanız gereken ardışık olarak sunulmuş olan yöntemlerden daima en alt seviyelerde yer alan yöntem veya algoritmayı seçmenizdir. Kullanacak olduğunuz algoritmayı seçtiğiniz takdirde seç düğmesinin yanında yer alan satırda seçtiğiniz yöntem veya algoritmaya ilişkin parametreler görülecektir. Eğer bu satırın üzerini çift tıklayacak olursanız J4.8 sınıflandırıcı editörü açılacak ve parametrelerin ne olduğu ve bunların program tarafından kabul edilen istenen değerlerinin hangi sayısal değerlere karşılık geldiği görülecektir. Tabi ki bu varsayılan değerler daha hassas ölçümler yapabilmek için belirlenmiştir.

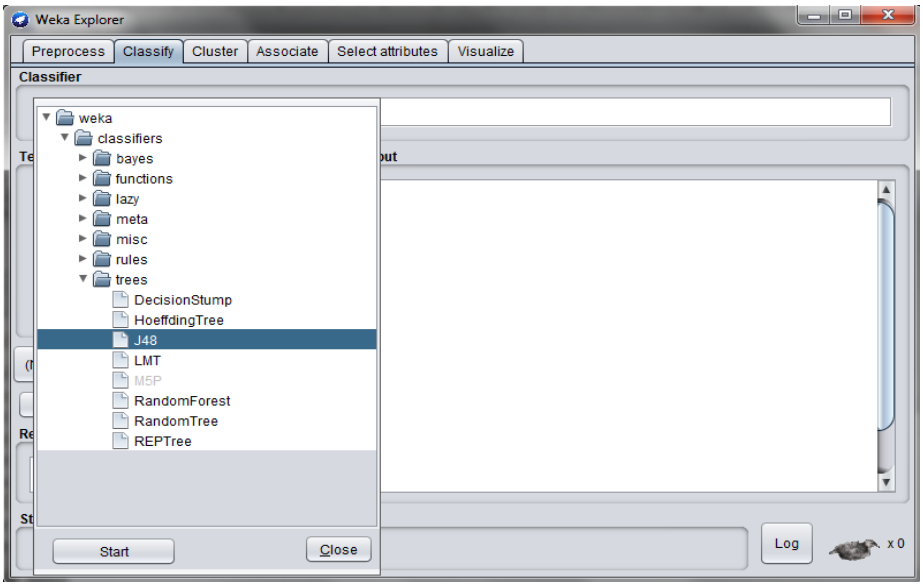
Sınıflandırıcıyı seçtikten sonra, Başlat (*Start*) düğmesini tıklayarak programı çalıştırabilirsiniz. WEKA programı oldukça hızlı bir işlem yeteneğine sahip olduğundan kısa sürede analiz yapabilmektedir ve analiz işlemi devam ederken Şekil 5 (a)'da gösterilen ana ekranın sağ alt köşesinde bir kuş hareket edecektir.

Örnek olması bakımından PISA 2015 öğrenci anketinde yer alan fen öğrenme süresi (*smins*), okulda toplam öğrenme süresi (*tmins*) ve öğrencinin sosyo-ekonomik durum indeksi değişkenleri (*escs*) ile fen okuryazarlık düzeylerinin (*pv1scie*) tahmin edilmesi sürecindeki girdi ve çıktı değişkenlerinin WEKA programına aktarıldıktan sonra elde edilen ekran görüntüsü Şekil 6'da gösterilmiştir.



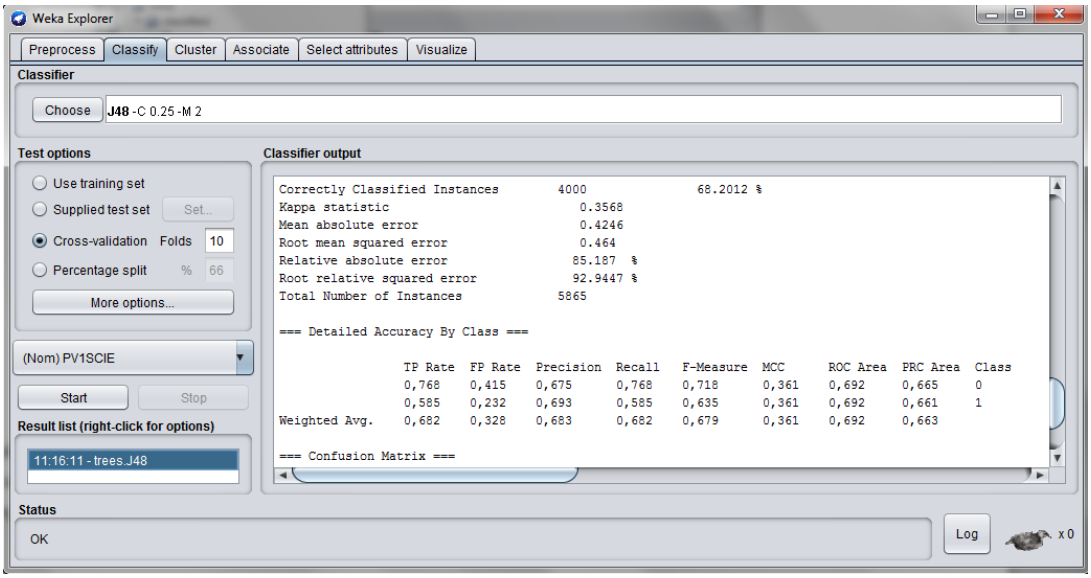
Şekil 6. Girdi ve Çıktı Değişkenlerinin Programa Aktarılması

Bu işlemin ardından Sınıflama (*Classify*) arayüzü açıldıktan sonra *Classifier* penceresindeki Seç (*Choose*) düğmesi tıklanarak Şekil 7'de gösterilen pencere ulaşılır ve Ağaçlar (*trees*) sekmesi altında yer alan karar ağaçlarından biri seçilir. Örnek olması bakımından J.48 yöntemi seçilerek analizler gerçekleştirilmiştir.



Şekil 7. Karar Ağacı Yönteminin Belirlenmesi

Şekil 7'de gösterilen ekranda Başlat (*Start*) düğmesi tıklanarak analiz işlemi tamamlanır. Eğer araştırmacılar 10 katlı çapraz geçerleme yöntemi yerine farklı bir yöntem kullanmak isterse test seçenekleri (*test options*) penceresindeki dört farklı geçerleme türünden bir tanesi seçilebilir. Bu işlemin ardından Başlat düğmesi tıklanarak Şekil 8'de gösterilen pencereye ulaşılır.

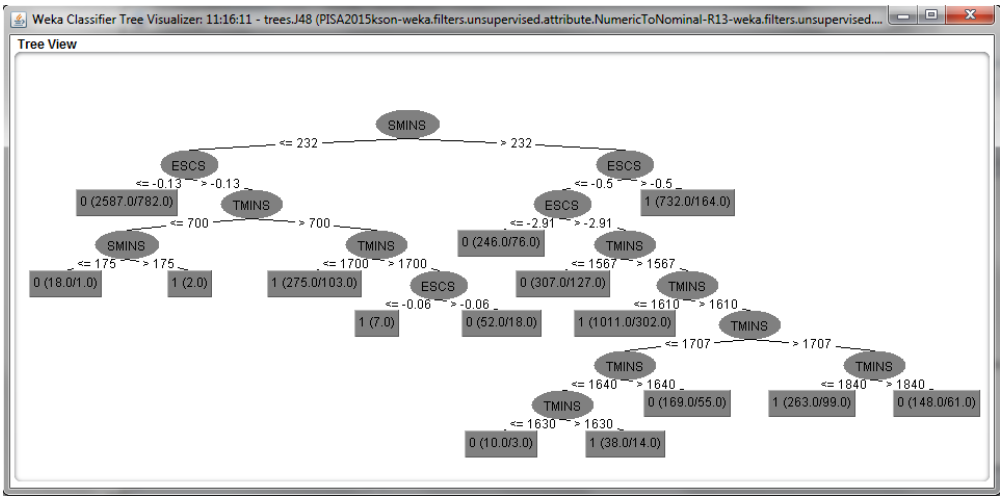


Şekil 8. Karar Ağacı Analiz Sonuçları

Şekil 8'de sınıflama sonuçları (*classifier output*) penceresinde toplam kaç öğrencinin doğru olarak sınıflandığı, sınıflama işlemine ilişkin istatistikler ve karışıklık (*confusion*) matrisi verilmektedir. Ekrandaki imleç aşağı doğru indirildiğinde sınıflama işlemine ilişkin diğer sonuçlar da görülebilecektir. Araştırmacıların karar ağacını görmek için Şekil 8'de gösterilen ekranın sol alt köşesinde yer alan Sonuçlar listesi (*Results list*) penceresindeki *trees.J48* satırının üzerine gidip sağ tıklayarak Ağacı görselleştir (*Visualize tree*) seçeneğini aktif hale getirmeleri gerekmektedir.

Sonuçların Değerlendirilmesi

Çalışma kapsamında PISA 2015 öğrenci anketinde yer alan fen öğrenme süresi (*smins*), okulda toplam öğrenme süresi (*tmins*) ve öğrencinin sosyo-ekonomik durum indeksi değişkenleri (*escs*) girdi; fen okuryazarlık düzeylerinin (*pv1scie*) çıktı değişkeni olarak tanımlandıktan sonra girdi değişkenleri yardımıyla çıktı değişkenin tahmin edilmesine ilişkin karar ağacı Şekil 9'da gösterilmiştir.



Şekil 9. J.48 Yöntemiyle Elde Edilen Karar Ağacı

Şekil 9'da görüldüğü üzere öğrencileri PISA Fen okuryazarlığı bakımından sınıflamak için çalışma kapsamında ele alınan üç girdi değişkeninden en önemli etkiye fen öğrenme süresi (*smins*) değişkeninin sahip olduğu görülmektedir. Öğrenciler fen öğrenme süreleri bakımından kesme puanı olarak belirlenen 232 değerine göre iki sınıfa ayrılmaktadırlar. Fen öğrenme süresi 232 değerinin altında olan öğrencileri sınıflamada en önemli etkiye sırasıyla sosyo-ekonomik durum indeksi ve toplam öğrenme süresi değişkenleri sahiptir. İkinci düzeydeki ağacın dallanmasında sosyo-ekonomik durum indeksine göre kesme puanı 0,13 olarak belirlenmiş ve bu değerin altında olan öğrenciler Başarısız (0) olarak sınıflanmışlardır. Aynı dal üzerinde sosyo ekonomik durum indeksi 0,13'ün üzerinde olanların toplam öğrenme süresine bakılmakta ve bu özellik için kesme puanı 700,00 olarak belirlenmektedir.

Çıktı dosyasında öğrenme yönteminin sınıflama hatasının yanı sıra Kappa istatistiği, ortalama mutlak hata ve ortalama hataların karekökü gibi ağaç tarafından sınıflama işlemine ilişkin değerlendirme sonuçları rapor edilmektedir. Ortalama hataların karekökü ikinci dereceden kayıp fonksiyonunun kareli ortalamasıdır. Ortalama mutlak hata benzer şekilde farkların hatalarını almak yerine bunların mutlak değeri alınarak hesaplanmaktadır. Aynı zamanda, çıktıda önsel olasılık dağılımlarına dayalı olarak hesaplanan bağıl hata değerleri de yer almaktadır. Bu istatistikler ZeroR öğrenme yöntemi kullanılarak elde edilmektedir. Son olarak çıktı dosyası incelendiğinde her bir sınıf için ayrıntılı doğruluk, hassaslık, geri çağırma ve F-ölçütü değerlerinin de rapor edildiği görülmektedir.

SONUÇLAR ve TARTIŞMA

Bir veri dosyasında var olan örüntüler arasından anlamlı sonuçlar çıkarmaya imkan sağlayan birçok farklı programdan biri olan WEKA programı kullanıcı dostu bir arayüze sahip olmasının yanında açık kaynak kodlu bir yazılım olması sebebiyle diğerlerinden daha popüler olarak görülmektedir (Zupan ve Demsar, 2008). Özellikle sayısal tahminde Bayesyen yöntemlerin yanında J4.8, ID3, M5P ve lojistik modele dayalı LMT algoritmaları size oldukça fazla seçenek sunmaktadır. Bunun yanında Random tree, Raptree ve Random forest gibi rastgele ağaç yapıları ile yüksek düzeyde doğruluk ve kesinlik değerine sahip kestirim yapmak mümkündür. Bağımsız değişkenler yardımıyla bağımlı değişkeni tahmin etmede klasik yöntemlerden bir adım daha ileri gidilerek tek bir regresyon denklemi vermek yerine lojistik regresyon modeli ile her bir dal için farklı denklemler yardımıyla yüksek doğruluk değerlerinde kestirimler yapılmaktadır (Robu ve Hora,2012). Yine programda kurallar ve fonksiyonlar menüsünün altında 20'nin üzerinde farklı seçenek sunulması programın önemli avantajları arasında yer almaktadır. Ayrıca programın .csv uzantılı veri dosyalarını herhangi bir dönüştürücüye gerek duymadan doğrudan okuyabilmesi önemli avantajları arasında yer almaktadır.

Her ne kadar Hoefding tree, J.48, Lojistik model, Reptree ve Random Tree algoritmaları için karar ağaçları program tarafından kolaylıkla oluşturulabilse de diğer algoritmalar için karar ağaçlarının verilmemesi programın sınırlıklarından biri olarak görülmektedir. Bunun yanında negatif işaretli ve ondalık değere sahip sayısal özelliklerin program tarafından doğrudan okunamaması bir sınırlılık olarak görülmektedir.

Veri madenciliği temel olarak sınıflama ve tahmin algoritmaları üzerine kuruludur. Her ne kadar sayısal bir özelliği belli bir eşik değerine göre iki sınıfa ayırdığınızda yine bir kategorik değişken elde etmiş olsanız da ikisi arasında önemli bir fark bulunmaktadır. Sınıflamalı özelliklerde dallanma sürecinde özelliğe ilişkin tüm bilgileri kullanmış olurken, sayısal özelliklerde ardışık bölünmemelerde özelliğe ilişkin bilgileri kullanmaya devam edersiniz. Yani sayısal özelliklerde ardışık dallanmalar yeni bilgiler üretmeye devam edecektir. Sınıflamalı bir özellik ağacın kökünden ağacın belli bir yaprağına kadar sadece bir kez test edilirken, sayısal bir özellik birçok kez test edilebilmektedir. Bu yüzden ağaçlar daha karmaşık ve anlaşılması daha zor olabilmektedir. Çünkü bir özelliğe ilişkin testler birlikte yapılmadığından yol boyunca dağılma olabilmektedir. Başarması daha zor ama daha okunabilir bir ağaç oluşturmanın yolu belirlenen özellik için çok yönlü test yapılmasına izin vermek ve ağacın tek bir düğümü üzerinde birkaç sabiti test etmektir. Bu sebeple sınıflama yerine sayısal özellikler üzerinden analizlere devam etmek daha faydalı görülmektedir. Ancak bu noktada WEKA programı ne yazık ki bağımsız değişkenler yardımıyla bağımlı değişkenin sayısal değerinin ne olacağını kestirememektedir. Her ne kadar gerekli kodlar yardımıyla öğrenme yöntemi tarafından kurulan lojistik model yardımıyla tek bir özelliğe ilişkin verileri modelde tanımladığınız taktirde ilgili örneğin sonuç değişkenini elde edebilmeniz mümkün olsa da çok büyük verilerin kullanıldığı veri madenciliğinde tüm örnekler için bağımlı değişkenin sayısal olarak tahmin edilememesi önemli bir kısıtlılık olarak görülmektedir.

Veri madenciliği konusunda çalışma yapacak araştırmacıların belirtilen üstünlükler ve sınırlılıklar çerçevesinde Waikato Üniversite tarafından ücretsiz olarak sunulan programın son versiyonlarından birini https://www.cs.waikato.ac.nz/ml/weka/downloading.html adresinden indirerek incelemeleri önerilmektedir. Özellikle programın örnek veri dosyaları üzerinden kendilerinin analizleri tekrarlayarak elde ettikleri sonuçları yorumlamaları önerilmektedir. Bunun yanında özellikle elinizde çok fazla değişken olması durumunda çapraz geçerleme yöntemine dayalı Özellikleri Seçme (Select Attributes) özelliğinin araştırmacıları değişken sayısını azaltma konusunda faydalı olacağı düşünülmektedir. Her ne kadar program açık kaynak kodlu olsa da özellikle bir programcı yardımıyla alan yazında çalışma yapan araştırmacıların kendilerinin yazdıkları komut dosyaları yardımıyla elde ettikleri sonuçları diğer yazılımlarla karşılaştırmaları önerilmektedir.

KAYNAKÇA

Bradley, E., & Tibshirani, R. J. (1993). *An introduction to the bootstrap*. Boca Raton, USA: Chapman and Hall.

Breiman L., Friedman J. H., Olsen E. A., & Stone C. J. (1984). *Classification and regression trees*. Belmont, CA: Wadsworth International Group.

Chadha, P., & Singh, G. N. (2012). Classification rules and genetic algorithm in data mining. *Global Journal of Computer Science and Technology Software & Data Engineering, 12*(15), 50-54.

Fayyad, U., & Irani, K. (1992) On the handling of continuous-valued attributes in decision tree generation. *Machine Learning, 8*, 87–102. doi: 10.1007/BF00994007

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). The KDD process of extracting useful knowledge from volumes of data. *Communications of the. ACM, 39*(11), 27–34. doi: 10.1145/240455.240464

Han J., Kamber, M., & Pei, J. (2000). *Data mining: Concepts and techniques*. Massachusetts: Morgan Kaufmann.

Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine Learning 11*, 63–91. doi: 10.1023/A:102263111

Ibarguren, I., Perez, J. M., Muguerza, J., Gurrutxaga, I., & Arbelaitz, O. (2014). *An update of the J48Consolidated WEKA's class: CTC algorithm enhanced with the notion of coverage*, (Technical Report EHU-KAT-IK-02-14), Spain: University of the Basque Country

Jain, A. K. (2015). *Indian ethnobotany: Emerging Trends*. Jodhpur: Scientific Publisher.

Johari, R. (2016). MS&E 226: "Small" data lecture 8: Classification problems, lucture notes, Retrived from: http://web.stanford.edu/~rjohari/teaching/notes.html

Karypis, G., Han, E. H., & Kumar, V. (1999). Chameleon: Hierarchical clustering using dynamic modeling. *Institute of Electrical and Electronics Engineers Computer Society*, *32*, 68-74. doi: 10.1109/2.781637

Kohavi, R. (1995, August) *A study of cross-validation and bootstrap for accuracy estimation and model selection*. Paper presented at the In Proceedings of International Joint Conference on AI. Montreal, Canada.

Kushary, D. (2012). Bootstrap Methods and Their Application, *Technometrics*, *42*(2), 216-217. doi: 10.1080/00401706.2000.10486018

Landis, J. R.,& Koch, G. G. (1977) The measurement of observer aggreement of categorical data. *Biometrics*, *31*(3), 159-174.

Lopez, M. I., Luna, J. M., Romero, C., & Ventura, S. (2012). *Classification via clustering for predicting final marks based on student participation in forums*. Paper presented at the 5th International Conference on Educational Data Mining, Chania, Greece.

Mitchell, T. M. (1997). *Machine learning*. New York: McGraw-Hill.

Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Mateo, CA: Morgan Kaufmann.

Padmavathi, J. (2012). Logistic regression in feature selection in data mining. *International Journal of Scientific & Engineering Research*, *3*(8), 1-4.

Perlich, C., Provost, F., & Simonoff, J. S. (2002). Tree induction vs. logistic Regression: A learning-curve analysis. The Journal of Machine Learning Research, *4*(1), 211-255. doi: 10.1162/153244304322972694

Powers, D. M. (2011). Evaluation: From precision, recall and F-measure to ROC, informedness, markedness & correlation. *Journal of Machine Learning Technologies*, *2*(1), 37–63. doi: 10.9735/2229-3981

Refaeilzadeh P., Tang L., & Liu, H. (2007). *On comparison of feature selection algorithms*. In Proc. AAAI-07 Workshop on Evaluation Methods in Machine Learing II. Vancouver, British Columbia, Canada, July 2007.

Robu, R., & Hora, C. C. (2012, June). *Medical data mining with extended WEKA*, Paper presented at the IEEE 16th International Conference on Intelligent Engineering Systems (INES), Lisbon, Portugal.

Rokach, L., & Maimon, O. Z. (2008). *Data mining with decision trees: Theory and applications*. Singapore: World Scientific Publishing Co., Inc.

Schwenke, C., & Schering, A. (2007). *True positives, true negatives, false positives, false negatives.* New Jersey, USA: Wiley Encyclopedia of Clinical Trials.

Sinha, A. P., & May, J. H. (2005). Evaluating and tuning predictive data mining models using receiver operating characteristic curves. *Journal of Management Information Systems*, *21*(3), 249-280.

Sumathi, S., & Sivanandam, S. N. (2006). *Introduction to data mining principles*. Berlin: Springer-Verlag.

Şeker, S. E. (2016). *Weka ve veri madenciliği*, Retrieved from https://www.dr.com.tr/ekitap/weka-ile-veri-madenciligi

Weiss, G. M., & Davison, B. (2010). Data mining, In H. Bidgoli (Ed.), *The handbook of technology management* (pp.2-17), New Jersey: John Wiley and Sons.

Witten, I. H., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques*. San Francisco: Morgan Kaufmann.

Witten, I. H., Frank, E., & Hall, M. (2016). *Data minig: Practical machine learning tools and techniques*. San Francisco: Morgan Kaufmann.

Zupan, B., & Demsar, J. (2008). Open-source tools for data mining, *Clinics in Laboratory Medicine, 28(1)*, 37-54. doi: 10.1016/j.cll.2007.10.002