Marmara Üniversitesi Öneri Dergisi • Cilt 14, Sayı 51, Ocak 2019, ISSN 1300-0845, ss. 90-111
DO1: 10.14783/maruoneri.vi.522168

Uygulamalı Makale

VERİ MADENCİLİĞİNDE SINIFLANDIRMA ALGORİTMALARININ PERFORMANS DEĞERLENDİRMESİ VE R DİLİ İLE BİR UYGULAMA*

PERFORMANCE EVALUATION OF CLASSIFICATION ALGORITHMS IN DATA MINING AND AN APPLICATION WITH THE R LANGUAGE

Ayşe ÇINAR**

Öz

Sınıflandırma Yöntemi, veri madenciliğinin başlıca yöntemlerinden biri olup, öğrenme algoritmasına dayanır. Büyük ölçekli bir veri içinde gizli kalmış bir örüntüyü keşfetmek amacıyla uygulanır. Veri madenciliği kapsamında, örüntü, bir varlık için dijital ortamda kaydedilmiş; gözlemlenebilir, ölçülebilir ve tekrar edilebilir bir bilgi olarak ifade edilmektedir. Ulaşılmak istenen bilginin elde edilmesi için uygulanan sınıflandırma algoritmaları, içerdiği verinin ortak özelliğine göre veri setinin belirli sınıflara ayrılmasını (ayrıklaştırılmasını) sağlamaktadırlar. Bu işlemin ardından bir sınıflandırma modeli elde edilir. Elde edilen sınıflandırma modeli yeni bir veri seti üzerinde uygulanarak, model ile belirlenmiş olan sınıfların veri seti içindeki benzerlerinin varlığı araştırılır. Söz konusu işlem "örüntü tanıma" olarak isimlendirilmektedir. Bu çalışmada, veri madenciliğinde sınıflandırma süreci ele alınarak, C5.0 ve Gini isimli iki farklı sınıflandırma algoritması ile bir uygulama gerçekleştirilmiştir. Bu amaçla açık kaynak kodlu R dili uygulanarak, her iki sınıflandırma modelinin tahmin değerlerinin doğruluğuyla ilgili performans ölçüm değerleri elde edilmiştir. Ayrıca, en iyi performans ölçüm değerine sahip bir model ele alınarak, sonuçları değerlendirilmiştir

Anahtar Kelimeler: Sınıflandırma Yöntemi, Sınıflandırma Algoritmaları, R Dili, Gini Algoritması, C5.0 Algoritması, Karışıklık Matrisi, Performans Değerlendirme.

JEL Kodları: C38, C8, I2

* Makale Gönderim Tarihi:27.11.2017; Makalenin Kabul Tarihi: 09.07.2018

** Marmara Üniversitesi, İşletme Fakültesi, İngilizce İşletme Bölümü, Dr. Öğr. Üyesi
ORCID ID: 0000-0001-7321-5959

Abstract

Knowledge discovery in databases (KDD) is the overall process of exploring previously unknown and useful knowledge in large volumes of data. The first stage of KDD is the process of ETL (extract, transform, load). It involves the following sequential steps in the process of KDD: Extracting raw data from a data source, applying data preprocessing and loading the processed data into several data repositories, such as databases, data warehouses. Data preprocessing technique is used to convert a raw data into a clean and proper data set according to the purpose of a related project. Data mining is an important part of the process in knowledge discovery. Compared to the traditional analyzing techniques, data mining is a process in order to extract understandable, valuable and previously unknown information in a large amount of dataset. Data mining techniques are divided into two different categories such as supervised learning and unsupervised learning. Supervised learning is a machine learning. Applying a supervised learning technique, a classification model called training model, is built with a reference. By using the built classification model, the class of testing data is predicted. Accordingly, there are some supervised learning techniques, such as Classification, Decision Tree, Bayesian Classification, Neural Networks, Association Rule Mining. Unsupervised learning is a type of machine learning. The difference between Supervised learning and Unsupervised learning is unsupervised learning learns from the data but without reference. Therefore, it is not necessary to create a prior model in unsupervised learning. Clustering is one of the unsupervised learning techniques. It separates data into some groups called clusters in which objects are similar to each other. Several data mining techniques have been developing that are used for knowledge discovery from a large amount of datasets including Classification, Clustering, Decision Tree, Bayesian Classification, Neural Networks, Association Rule Mining, Prediction, Sequential Pattern and Genetic Algorithm, Time Series and Nearest Neighbor. The classification method which is one of the main methods of data mining is based on learning algorithm. It is applied in order to discover hidden patterns in a large-scale data. Following the ETL process, a classification model is created by selecting one of data mining methods. Within the scope of data mining, a pattern is expressed as an observable, measurable and repeatable information that is stored in digital area for an entity. Classification algorithms that are applied in order to obtain a target information separate a dataset into several groups according to the common feature of the data. After the mentioned process, a classification model is obtained. Applying the obtained classification model on a new data set, the similar examples of the classes that are determined by the model are analyzed. The mentioned process is called as "pattern recognition". The dataset is divided into two sets called training and testing datasets in order to build predictive models. The aim of the study is to apply some classification algorithms on a dataset and evaluate the performance of the models in terms of the prediction accuracy. For the purpose of the study, a database named "Data_User_Modeling_Dataset_Hamdi_Tolga_KAHRAMAN.xls" was chosen as sample case. The database contains raw data about the knowledge level of the learners in e-learning systems. It is possible to download the mentioned data from the website named "UCI Machine Learning" as a dataset. In the study, an application was performed by two different classification algorithms called C5.0 and Gini by considering the classification process in data mining. Additionally, in order to build some predictive models, the dataset was divided into two different sets called training and testing datasets with predetermined rate in the whole dataset. Accordingly, the open-source R programming language was applied for the both classification algorithms in order to build a classification model. As a result of the execution of

the written R codes, some decision rules and a decision tree were obtained for both algorithms with the handled training dataset. After the prediction of the class of each testing data, the performance measures on the accuracy of the predicted values of the both models were estimated with the current class of each observation in the testing dataset. When the results were evaluated, a model that had the best performance was handled and its results were evaluated. The results of the selected classification model showed that the attribute related to the exam performance of the learners for goal objects (PEG) became the most deterministic predictor on their knowledge levels. Accordingly, the attribute related to the exam performance of the learners for related objects with goal object (LPR) took second place in order of importance.

Keywords: Classification Method, Classification Algorithms, R Language, Gini Algorithm, C.50 Algorithm, Confusion Matrix, Performance Evaluation.

JEL Codes: C38, C8, I2

1. GİRİŞ

Günümüz bilgi çağı ve gelişen teknolojiler, her alanda hızla artış gösteren ölçekte veri birikimine neden olmaktadır. Veri, bilgi ve bilgi yönetimi kavramları oldukça önemli hale gelmiştir. Karar destek sistemlerinin kullandığı araçlardan biri olan veri madenciliği, kısaca değeri olan bir bilgiye ulaşma süreci olarak adlandırılır. Bu özelliği nedeniyle, büyük ölçekli bir veriden anlamlı, uygulanabilir bilgi elde etmek amacıyla, her alanda uygulanma oranı giderek artış göstermektedir. İçerdiği yöntemler aracılığıyla, bilim ve teknoloji alanında oldukça önemli katkılar sağlamaktadır.

Veri madenciliği süreci çeşitli aşamalar içermekte olup, bu aşamalar ana başlıklar halinde aşağıda görüldüğü gibi listelenmektedir (Kantardzic, 2011):

- Veri madenciliğine konu olan problemin tanımı,
- Verinin elde edilmesi,
- Veri üzerinde düzeltme, eksik ya da hatalı veri temizleme, bütünleştirme ve dönüştürme gibi çeşitli ön işlemler yapılarak, verinin analize uygun hale getirilmesi,
- Veri madenciliği yöntemlerinin uygulama aşaması,
- Uygulanan yöntemlerden elde edilen sonuçların performans değerlendirmesinin yapılması,
- En iyi performansa sahip olan yöntemin sonuçlarının değerlendirilmesi.

Veri madenciliği yöntemleri üç ana başlık altında toplanmaktadır:

1. Sınıflandırma Yöntemi
2. Kümeleme Yöntemi
3. Birliktelik Analizi

Sınıflandırma Yöntemi: Tahmin edici, ya da bir diğer ifadeyle kestirimci bir yöntemdir (Han & Kamber, 2012). Sınıflandırma Yöntemi bir veri setinin, içerdiği verinin ortak özelliğine göre belirli sınıflara ayrılmasını sağlamaktadır. Bu amaçla çeşitli algoritmalar geliştirilmiştir. Başlıca algoritmalar, entropi tabanlı sınıflandırma (C4.5 algoritması, C5.0 algoritması), Regresyon ve Karar Ağaçları (Gini algoritması, Twoing algoritması), Bellek Tabanlı Algoritmalar (k-en yakın komşu algoritması), Bayes Sınıflandırıcılar, Regresyon Ağaçları, Rastgele Orman şeklinde sıralanabilir (Özkan & Erol, 2015).

Sınıflandırma yapılacak bir veri setinde yer alan her örnek çeşitli niteliklere sahiptir. Bu niteliklerin biri sınıf bilgisini veren ve *"çıktı"* olarak isimlendirilen bir hedef nitelik olup, diğer nitelikler *"girdi"* olarak isimlendirilirler. Uygulama yapılacak veri setinin bu özelliğinden dolayı, bu çalışmada uygulanacak karar ağaçları ile sınıflandırma yöntemi danışmanlı öğrenmeye dayalı bir yöntem olarak isimlendirilir (Cunningham, Cord & Delany, 2008). Dolayısıyla, yöntemin uygulanma aşamasında veri setinin belirli bir oranı rastgele seçilerek eğitim veri seti olarak ele alınır ve kalan kısmı ise test veri seti olarak değerlendirilir. Sınıflandırma modeli elde edilecek bir eğitim veri setinde hangi sınıfa ait olduğu bilinen örnekler yer almaktadır. Söz konusu veri seti üzerinde sınıflandırma algoritmalarından biri uygulanarak bir sınıflandırma modeli oluşturulur ve ardından test veri setindeki örnekler için sınıf tahmini yapılarak, modelin tahmin sonucunun performans ölçüm değeri elde edilir.

Bu çalışma kapsamında sınıflandırma algoritmaları içinden karar ağacı ve karar kuralları oluşturan C5.0 ve Gini algoritmaları ele alınmıştır. C5.0 algoritması, ikili ya da daha fazla bölünmeye dayalı bir sınıflandırma algoritması olup, bölünme ölçütü olarak bilgi kazancı (information gain) değeri ele alınır. Söz konusu algoritma ile her bir karar noktasından ya da bir diğer ifadeyle karar düğümünden itibaren iki ya da daha fazla dala ayrılan bir karar ağacı geliştirilir. Karar ağacının içerdiği karar düğümlerinin belirlenmesi aşamasında, entropi hesabı yapılır ve buna göre eğitim veri seti içinde girdi olarak tanımlanan niteliklerin bilgi kazançları belirlenir. Bu işlemin ardından, en yüksek kazanca sahip olan nitelik ile karar düğümü oluşturulur (Pandya & Pandya, 2015). C5.0 algoritmasının uygulanabilmesi için sınıflandırma yapılacak hedef niteliğin kategorik olması gerekir. Gini algoritması için böyle bir kural söz konusu olmayıp, hedef niteliğin sayısal ya da kategorik türleri için uygulanabilir. Gini algoritması ile her karar düğümü ikiye ayrılan Sınıflandırma ve Regresyon Ağaçları (CART) oluşturulur (Adak & Yurtay, 2013). İkili bölünmeye dayalı olan Gini algoritmasının bölünme ölçütü olarak Gini indeks değeri ele alınır (Kumar & Kiruthika, 2015). Gini bölünme (gini split) olarak da isimlendirilen bu indeks değeri, karar ağacı oluşumunun her aşamasında veri setinin girdi olarak tanımlanan bütün nitelikleri için hesaplanır ve en düşük Gini indeks değerine sahip olan nitelik ile karar düğümü oluşturulur (Özkan, 2016). C5.0 ve Gini algoritmaları aracılığıyla geliştirilen karar ağaçları ise karar kurallarının oluşmasını sağlar.

Kümeleme Yöntemi: Çeşitli uzaklık yöntemleri aracılığıyla, bir veri seti içinde birbirine benzer örnekleri kümelere ayırma işlemidir (Hastie, Tibshirani & Friedman, 2008).

Kümeleme yapılacak bir veri setinde, örnekler için sadece *"girdi"* nitelikler yer alır. Bu özelliği ile danışmansız öğrenmeye dayalı bir yöntem olarak ele alınan kümeleme yöntemi ile bir çıktı değeri elde etmeksizin, girdi değerler arasındaki ilişki ve örüntüler tanımlanır. Hiyerarşik ve Hiyerarşik olmayan kümeleme yöntemleri mevcuttur. Hiyerarşik küme yöntemleri için geliştirilen başlıca algoritmalar k-en yakın komşu ve k-en uzak komşu algoritmalarıdır. K-ortalamalar ise Hiyerarşik olmayan kümeleme yöntemlerinden biridir.

Birliktelik Kuralları: Birliktelik analizi yapılacak verinin her satırı araştırma konusuna göre; bir olay, eylem, ya da müşteri şeklinde ele alınır. Bu yöntem, veri seti içindeki eylemlerin birlikte gerçekleşme durumlarını ortaya koymaktadır. Bunun için "destek" ve "güven" ölçütleri kullanılır. Birliktelik kuralları için geliştirilen algoritmalar için en yaygın kullanılan algoritma Apriori Algoritması'dır (Han & Kamber, 2012).

Araştırmanın amacı C5.0 ve Gini isimli iki farklı sınıflandırma algoritması ile bir uygulama gerçekleştirmektir. Elde edilen sonuçlar üzerinde her iki algoritma için elde edilen karışıklık matrisi (confusion matrix) doğrultusunda performans değerlendirmesi yapılmıştır. Algoritmalar içinde en iyi performans ölçüm değerine sahip olduğu görülen C5.0 algoritmasının sonuçları ele alınmıştır. Söz konusu algoritmaya ait karışıklık matrisi değerlerinin hesaplanma yöntemi ifade edilmiş ve ayrıca, karar ağaçları oluşturularak, karar kuralları geliştirilmiştir.

2. SINIFLANDIRMA ALGORİTMALARININ R DİLİ ile UYGULANMASI

2.1. Veri Hazırlama Süreci

Sınıflandırma algoritması uygulaması aşamasında UCI Machine Learning Repository sitesinde yer alan; öğrencilerin öğrenme düzeylerinin sınıflandırılmasına yönelik olan *"Data_User_Modeling_Dataset_Hamdi Tolga KAHRAMAN.xls"* isimli veri dosyası indirilmiştir (UCI, 2009). Bu dosya, sınıflandırma sürecinde kullanmak amacıyla, *".csv"* uzantılı olarak kaydedilmiştir. Dosya içindeki bilgiler; Gazi Üniversitesi Teknik Eğitim Fakültesi Elektrik Eğitimi Bölümü lisans öğrencilerine aittir. Bu bilgiler, öğrenciler ya da bir diğer ifadeyle kullanıcılar için yeterli veya uygun bilgi sağlayan bir kullanıcı modelleme sistemi aracılığıyla elde edilmiştir. Kullanıcı modeller çoğunlukla web tabanlı uygulamalar olup, özellikle çevrimiçi öğrenme ortamları için oluşturulmuştur. Söz konusu çalışmada, bu sistem aracılığıyla, öğrencilerin eğitim faaliyetleri hakkında; ziyaret edilen ders sayfaları, tuş vuruşları, kavram/amaç sayfalarında geçirdikleri süre ve sınav performansları ve benzeri bilgiler kaydedilmiştir (Kahraman, Sağıroğlu & Çolak, 2013). Ele alınan veri seti içerisinde toplam altı adet nitelik yer almakta olup, bu nitelikler ile ilgili açıklamalar aşağıda ve ayrıca detaylı olarak Tablo1'de görülmektedir.

STG: Sistemi kullanan öğrencilerin öğrenilecek bir konu ile ilgili materyaller ile çalışma süresinin derecesi.

SCG: Öğrencilerin öğrenilen materyalleri tekrar etme sayısının derecesi.

STR: Öğrencilerin öğrenilecek bir konu ile ilgili olan diğer konuları çalışma süresinin derecesi.

LPR: Öğrencilerin öğrenilecek konu ile ilgili diğer konular için yapılan sınavdaki performansı.

PEG: Öğrencilerin hedef konular için yapılan sınavdaki performansı.

UNS: Öğrencilerin bilgi düzeyi.

Veri içindeki toplam kayıt sayısı 431 olup, hedef niteliğe (UNS) ait sınıfların toplam kayıt dağılımı şu şekildedir; Very Low: 50 kayıt, Low: 129 kayıt, Middle: 122 kayıt ve High: 130 kayıt.

Tablo1: Veri Seti İçerisindeki Niteliklerin Özellikleri

Nitelik Adı	Açıklama	Veri Tipi	Nitelik Tipi
STG	Öğrenim materyalleri ile çalışma süresinin derecesi	Nümerik	Girdi
SCG	Öğrenilen materyalleri tekrar etme sayısının derecesi	Nümerik	Girdi
STR	Konu ile ilgili olan diğer konuları çalışma süresinin derecesi	Nümerik	Girdi
LPR	Diğer konular için yapılan sınavdaki performans	Nümerik	Girdi
PEG	Hedef konular için yapılan sınavdaki performans	Nümerik	Girdi
UNS	Öğrencilerin bilgi düzeyi	Kategorik	Çıktı (hedef)

R dilinde veri içindeki niteliklerin özelliklerini incelemek için *str()* fonksiyonu kullanılır. Bu fonksiyonun çalıştırılması sonucu elde edilen sonuç aşağıdaki gibidir.

```
> str(x)
                    'data.frame': 403 obs. of  6 variables:
                  $ STG: num  0 0.08 0.06 0.1 0.08 0.09 0.1 0.15 0.2 0 ...
                  $ SCG: num  0 0.08 0.06 0.1 0.08 0.15 0.1 0.02 0.14 0 ...
                  $ STR: num  0 0.1 0.05 0.15 0.08 0.4 0.43 0.34 0.35 0.5 ...
                  $ LPR: num  0 0.24 0.25 0.65 0.98 0.1 0.29 0.4 0.72 0.2 ...
                  $ PEG: num  0 0.9 0.33 0.3 0.24 0.66 0.56 0.01 0.25 0.85 ...
                  $ UNS: Factor w/ 4 levels "High","Low","Middle",..: 4 1 2 3 2 3 3 4 2 1
```

Şekil 1: Veri Setinin *Str()* Fonksiyonu ile Elde Edilen Görünümü

Ekran görünümünde, diğer niteliklerin nümerik ve UNS isimli niteliğin kategorik değere sahip olduğu görülmektedir. Bu nitelik dört farklı kategorik değere sahiptir; *"High"*, *"Low"*, *"Middle" ve "Very Low"*. Söz konusu değerler sınıflandırma aşamasında hedef nitelik olarak ele alınacak olan UNS niteliğinin dört farklı sınıfı olarak değerlendirilecektir.

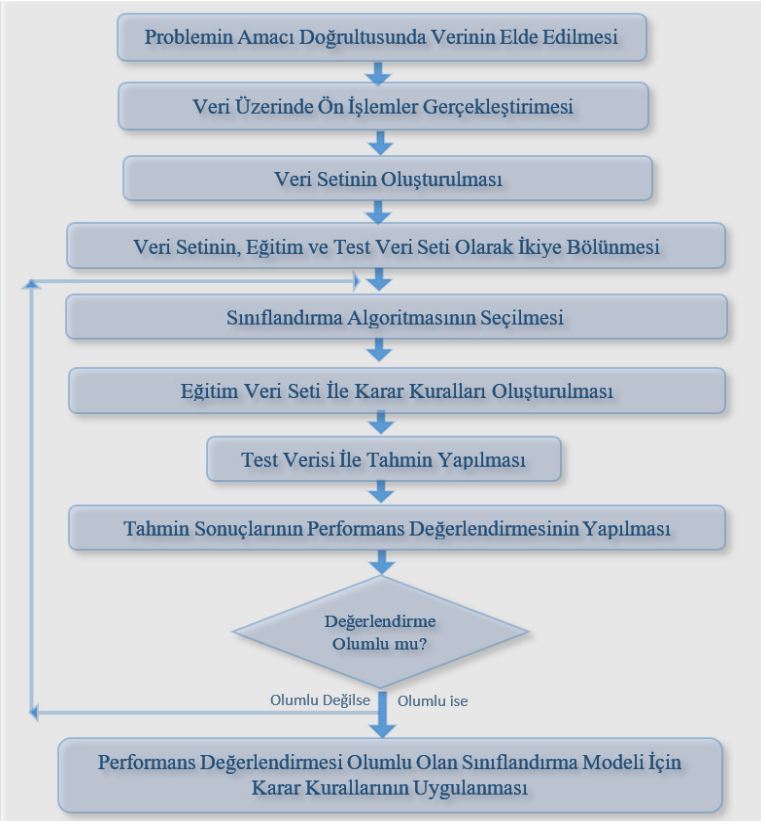
Analiz öncesi veri üzerinde gerekirse ön işlemler uygulanır. Bu aşamada, verinin analiz için uygun olup olmadığı kontrol edilir. Veri içerisinde eksik veri, ya da aykırı veri (outliers) bulunabilir. Eğer eksik, doğru ya da analiz için yeterli olmayan verinin varlığı saptanırsa, veri

üzerinde ön işlemler yaparak, bu sorunlar giderilir. Böyle bir durumda, eksik verinin yerine ortalama bir değer ya da farklı tahmin yöntemleri ile elde edilen bir değer yerleştirilerek sorunun giderilmesi gerekir. Aykırı verinin varlığı söz konusu ise, Min-Max Normalizasyon ya da Zscore Standardizasyon gibi Normalizasyon tekniklerinden biri uygulanmalıdır. Eğer başka bir aralık tanımlanmazsa aykırı değerler, bu teknikler aracılığıyla 0-1 aralığında ölçeklenir.

Bu araştırmada uygulanacak veri dosyasının içerdiği verinin sınıflandırma yöntemlerinin uygulanmasına hazır halde olduğu belirlenmiştir. Bu nedenle, veri üzerinde herhangi bir ön işlem yapılmasına gerek olmaksızın, bu çalışmada veri seti olarak değerlendirilmiştir.

2.2. Veri Madenciliği

Veri madenciliği aşamasında, C5.0 ve Gini algoritmaları aracılığıyla iki sınıflandırma modeli geliştirilmiştir. Her iki model benzer bir sınıflandırma sürecine sahip olup, bu süreç aşamalar halinde Şekil 2'de ifade edilmiştir. Bu çalışma kapsamında, sınıflandırma sürecindeki söz konusu işlemler sırasıyla gerçekleştirilecektir.



Şekil 2: Sınıflandırma Süreci

2.2.1. Veri Oluşturma

UCI Machine Learning Repository isimli web sitesinde veri madenciliği yöntemlerinde uygulanması amacıyla çeşitli konularda veri setleri mevcut olup, araştırmacıların kullanıma sunulmaktadır. Bu sitedeki; öğrencilerin öğrenme düzeylerinin sınıflandırılmasına yönelik olan *"Data_User_Modeling_Dataset_Hamdi Tolga KAHRAMAN.xls"* isimli bir veri dosyası indirilerek, sınıflandırma sürecinde kullanmak amacıyla, *".csv"* uzantılı olarak kaydedilmiştir.

Bu dosya aşağıda verilen komut dizilimi aracılığıyla, R çalışma ortamına aktarılmıştır.

```
veri<-read.csv("Data_User_Modeling_Dataset.csv",header = TRUE,sep =";",dec=",")
head(veri)
```

Şekil 3: Verisetini Çalışma Ortamına Aktarma ile İlgili Komut Dizilimi

Komut diziliminde, *".csv"* uzantılı bir dosyaya erişmek için *read.csv()* fonksiyonu kullanılmıştır. Bu fonksiyonun parametreleri aracılığıyla, veri seti içinde, her niteliğe ait bilgilerin noktalı virgül ile ayrıldığını (*sep =";"*) ve sayıların ondalık işaretinin virgül olduğu (*dec=","*) belirtilmiştir. Bu işaretler, bir veri setinin içerdiği veri yapısına göre değişebilir ve dolayısıyla, uygulanacak bir veri setinin işaretlerinin özelliğine uygun olarak parametre değerleri atanmasına dikkat edilmelidir. Aksi bir durumda veri R çalışma ortamına aktarılamaz.

Head() fonksiyonu ile veri setinin ilk altı satırı görüntülenmektedir (Şekil 4).

```
> head(veri)
  STG  SCG  STR  LPR  PEG      UNS  X X.1
1 0.00 0.00 0.00 0.00 0.00 Very Low NA  NA
2 0.08 0.08 0.10 0.24 0.90     High NA  NA
3 0.06 0.06 0.05 0.25 0.33      Low NA  NA
4 0.10 0.10 0.15 0.65 0.30   Middle NA  NA
5 0.08 0.08 0.08 0.98 0.24      Low NA  NA
6 0.09 0.15 0.40 0.10 0.66   Middle NA  NA
```

Şekil 4: Veri Setinin İlk Altı Satırı

Çalışma ortamına aktarılan veri setinin son iki sütunu analiz aşamasında kullanılmayacağı için modelden çıkarılacaktır. Bu amaçla, indirilmiş olan veri setinin ilk altı sütununu içeren *"eksik veri"* isimli yeni bir veri seti oluşturulmuş ve ismi *"x"* olarak değiştirilmiştir. Bu işlemler ve veri setinin son şekli aşağıda görülmektedir.

```
> eksik_veri<-veri[,c(1:6)]
> x<-eksik_veri
> head(x)
  STG  SCG  STR  LPR  PEG  UNS
1 0.00 0.00 0.00 0.00 0.00 Very Low
2 0.08 0.08 0.10 0.24 0.90 High
3 0.06 0.06 0.05 0.25 0.33 Low
4 0.10 0.10 0.15 0.65 0.30 Middle
5 0.08 0.08 0.08 0.98 0.24 Low
6 0.09 0.15 0.40 0.10 0.66 Middle
```

Şekil 5: Yeni Veri Seti

2.2.2. Eğitim ve Test Veri Setleri Oluşturma

Veri seti analize hazır durumda olduğu için veri ön işlemleri yapılmaksızın, eğitim ve veri setleri oluşturulmuştur. Bu işlem için *"caret"* paketinin yüklenmesi ve kütüphanesine erişilmesi gerekir (Kuhn, 2017). Komut dizilimi aşağıda görülmektedir.

```
install.packages("caret")
library(caret)
```

Şekil 6: *"caret"* Paketinin Yüklenmesi ile İlgili Komut Dizilimi

"caret" paketinin içinde yer alan *CreateDataPartition()* fonksiyonu ile veri seti ikiye ayrılır. *"ayrım"* isimli değişkene atanan bu işlem aşamasında *"y"* parametresi ile *"x"* isimli veri setinin hedef niteliğinin *"UNS"* niteliği olduğu bildirilir (y= x$UNS). Bu fonksiyonun bir diğer parametresi de *"p"* parametresi olup, bu parametreye ayrımın yüzdesi aktarılır. Aşağıdaki kod diziliminde bu değerin %75 olacağı belirtilmiştir (p=.75). Bu oran eğitim veri seti için verilmiştir. Bu nedenle, bu işlemin atandığı *"ayrım"* isimli değişkenin içeriği *"egitim"* isimli bir değişkene aktarılır. Veri setinin geri kalan %25'i test veri seti olacak şekilde (test<-x[-ayrım,]) aşağıda görüldüğü gibi kod yazılır. Test verisi için yazılan bu kod diziliminde, parantez içinde *"ayrım"* dan önce "-" işaretinin olması gerekir. Ayrıca, *set.seed()* fonksiyonuna bir başlangıç değeri verilerek, her defasında aynı kayıtlara sahip olacak bir ayrımın elde edilebilmesi sağlanır (Balaban & Kartal, 2016).

```
set.seed(1)
ayrım< – createDataPartition(y = x$UNS, p = .75, list = FALSE)
egitim<-x[ayrım,]
test<-x[-ayrım,]
```

Şekil 7: Veri Setinin Bölünmesi ile İlgili Komut Dizilimi

Hedef niteliğin (*UNS*), veri setindeki dizin numarası 6'dır. Bu dizin numarası x[[6]] şeklinde kullanılarak, eğitim ve test verileri için *"egitim_hedef"* ve *"test_hedef"* değişkenlerine

atanır. Bu işlem sayesinde, söz konusu değişkenler sadece UNS sütununa sahip olan bir vektör haline gelmişlerdir.

```
egitim_hedef<-x[[6]]
test_hedef<-x[[6]]
```

Şekil 8: Eğitim ve Test Veri Setlerinin Oluşturulması ile İlgili Komut Dizilimi

Eğitim ve test veri setleri için hedef niteliğin dışındaki nitelikler, *"egitim_nitelikler"* ve *"test_nitelikler" isimli* değişkenlere aktarılır. Kod diziliminde dizin numarasından önce "-" yazılarak (x[,-6]), bu durum ifade edilmiştir.

```
egitim_nitelikler<-x[,-6]
test_nitelikler<-x[,-6]
```

Şekil 9: Hedef Nitelik Hariç Diğer Niteliklerin Tanımlanması ile İlgili Komut Dizilimi

2.2.3. Sınıflandırma Yöntemlerinin Uygulanması

Önceki aşamada oluşturulan eğitim ve test veri setlerini kullanarak, sınıflandırma işlemi gerçekleştirilir. Bu çalışmada kullanılan veri setinin hedef niteliği kategorik değere sahiptir. Bu nedenle, sınıflandırma yapmak amacıyla, sınıflandırma algoritmaları içinde kategorik değere sahip olan bir hedef nitelikle çalışabilen Gini ve C5.0 algoritmaları seçilmiştir.

2.2.3.1. Gini Algoritması ile Sınıflandırma

Gini algoritması ile sınıflandırma yapmak için *"rpart" paketinin* yüklenmesi ve kütüphanesine erişilmesi gerekir. Bu paket içinde yer alan *rpart()* fonksiyonu ile sınıflandırma işlemi gerçekleştirilir (Therneau, Atkinson & Ripley, 2017).

```
install.packages("rpart")
library(rpart)
gini<-rpart(UNS ~., data=egitim,method="class",minsplit=4,parms=list(split="gini"))
show(gini)
```

Şekil 10: Gini Algoritması ile Sınıflandırma İşlemi ile İlgili Komut Dizilimi

rpart() fonksiyonu aracılığıyla, sınıflandırma kuralları ve karar ağacı oluşturulmuştur. *show()* fonksiyonu ile karar kuralları görüntülenebilir (Şekil 11).

```
show(gini)

n= 304
node), split, n, loss, yval, (yprob).
    * denotes terminal node
.
1) root 304 207 Low (0.253289474 0.319078947 0.302631579 0.125000000)  .
  2) PEG>=0.39 155  78 High (0.496774194 0.006451613 0.496774194 0.000000000)  .
    4) PEG>=0.685 69   1 High (0.985507246 0.000000000 0.014492754 0.000000000) *.
    5) PEG< 0.685 86  10 Middle (0.104651163 0.011627907 0.883720930 0.000000000)  .
     10) LPR>=0.805 8   0 High (1.000000000 0.000000000 0.000000000 0.000000000) *.
     11) LPR< 0.805 78   2 Middle (0.012820513 0.012820513 0.974358974 0.000000000) *.
  3) PEG< 0.39 149  53 Low (0.000000000 0.644295302 0.100671141 0.255033557)  .
    6) PEG>=0.135 110  21 Low (0.000000000 0.809090909 0.136363636 0.054545455)  .
     12) LPR< 0.79 98  12 Low (0.000000000 0.877551020 0.061224490 0.061224490) *.
     13) LPR>=0.79 12   3 Middle (0.000000000 0.250000000 0.750000000 0.000000000) *.
    7) PEG< 0.135 39   7 Very Low (0.000000000 0.179487179 0.000000000 0.820512821)  .
     14) LPR>=0.62 8   1 Low (0.000000000 0.875000000 0.000000000 0.125000000) *.
     15) LPR< 0.62 31   0 Very Low (0.000000000 0.000000000 0.000000000 1.000000000) *
```

Şekil 11: Gini Algoritması ile Elde Edilen Karar Kuralları

Bir sonraki aşama, test verisini kullanarak, Gini algoritması ile oluşturulan karar kuralları doğrultusunda bir tahmin yapılmasıdır. Aşağıdaki kod dizilimi bu işlemi yapmaktadır.

```
tahmin_Gini < – predict(gini,test,type="class")
```

Şekil 12: Gini Sınıflandırma Modeli ile Tahmin Gerçekleştirilmesi ile İlgili Komut Dizilimi

2.2.3.1.1. Gini Algoritması İçin Performans Değerlendirme (Karışıklık Matrisi)

Test verisi ile yapılan test aşamasında elde edilen tahmin değerlerinin ne derece doğru sonuçlar ürettiğini görmek için *confusionMatrix()* fonksiyonu kullanılır (Kohavi & Provost, 1998). Bu fonksiyonu kullanmak için "*caret" paketinin* yüklenmiş ve kütüphanesine erişilmiş olması gerekmektedir (Kuhn, 2017). Söz konusu fonksiyon aracılığıyla, test verisinin hedef niteliği ile tahmin sonuçları karşılaştırılır. Bunun için, kod diziliminde *reference* parametresine *test[,6]* ifadesi atanır.

```
cm_gini < – confusionMatrix(data = tahmin_Gini, reference = test[,6])
cm_gini
```

Şekil 13: Gini Sınıflandırma Modelinin Performans Değerendirmesi ile İlgili Komut Dizilimi

Bu işlemin sonunda karışıklık matrisi ve bu matris aracılığıyla hesaplanan performans ölçüm değerleri elde edilir (Sokolova & Lapalme, 2009). Şekil 14'deki söz konusu performans

ölçüm değerlerine bakıldığında, modelin doğruluk oranının *accuracy= 0.9394* olduğu görülmektedir. Oldukça yüksek bir oran olup, bu değer doğru tahmin edilen sınıfların sayısının, toplam tahmin sayısına olan oranını ifade etmektedir *(doğru tahmin/toplam tahmin).*

```
> cm_gini
Confusion Matrix and Statistics

          Reference
Prediction High Low Middle Very Low
  High       24   0      0        0
  Low         0  32      4        1
  Middle      1   0     26        0
  Very Low    0   0      0       11

Overall Statistics

               Accuracy : 0.9394
                 95% CI : (0.8727, 0.9774)
    No Information Rate : 0.3232
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.916
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: High Class: Low Class: Middle Class: Very Low
Sensitivity               0.9600     1.0000        0.8667          0.9167
Specificity               1.0000     0.9254        0.9855          1.0000
Pos Pred Value            1.0000     0.8649        0.9630          1.0000
Neg Pred Value            0.9867     1.0000        0.9444          0.9886
Prevalence                0.2525     0.3232        0.3030          0.1212
Detection Rate            0.2424     0.3232        0.2626          0.1111
Detection Prevalence      0.2424     0.3737        0.2727          0.1111
Balanced Accuracy         0.9800     0.9627        0.9261          0.9583
```

Şekil 14: Gini Algoritması ile İlgili Karışıklık Matrisi ve Performans Değerleri

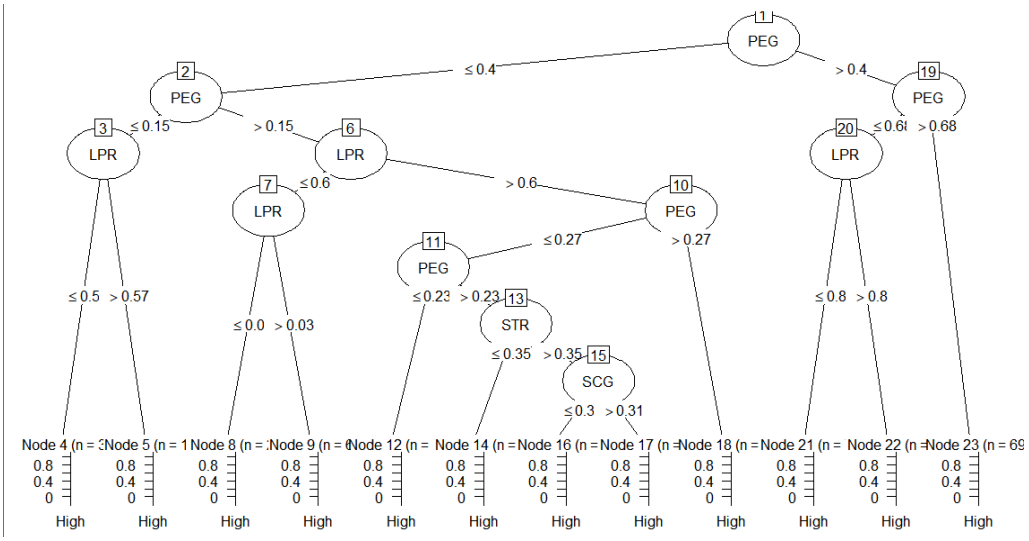
2.2.3.2. C5.0 Algoritması ile Sınıflandırma

C5.0 algoritması ile sınıflandırma yapmak için *"C50" paketinin* yüklenmesi ve kütüphanesine erişilmesi gerekir (Kuhn, Weston, Coulter, Culp & Quinlan, 2018). Bu paket içinde yer alan *C5.0()* fonksiyonu ile sınıflandırma işlemi gerçekleştirilir. *plot()* fonksiyonu ile karar ağacı çizilir ve Gini algoritmasında olduğu gibi *predict()* fonksiyonu ile test verisi kullanarak tahmin gerçekleştirilir. Bu işlemlerin ardından *confusionMatrix()* fonksiyonu ile performans değerlendirme yapılır. Bu işlemden önce "*caret" paketinin* yüklenmiş ve kütüphanesine erişilmiş olması gerekmektedir (Kuhn, 2017).

```
install.packages("C50")
library(C50)
C50 <-C5.0(UNS ~., data=egitim)
summary(C50)
plot(C50)
tahmin_c50 < – predict(C50,test,type="class")
tahmin_c50
test[,6]
# Karışıklık Matrisi Elde Edilmesi
cm_c50 < – confusionMatrix(data = tahmin_c50, reference = test[,6])
cm_c50
```

Şekil 15: C5.0 Algoritması ile İlgili Komut Dizilimi

Elde edilen sonuçlar sırasıyla Şekil 16 ve Şekil 17'de görülmektedir.



Şekil 16: C5.0 Algoritması ile İlgili Karar Ağacı

C5.0 algoritması için elde edilen karışıklık matrisi ve performans ölçüm değerlerine bakıldığında, modelin tahmin değerlerinin doğruluk oranının *accuracy=0.9596* olduğu görülmektedir (Şekil 17). Bu değer Gini algoritması ile elde edilen doğruluk değerinden daha yüksektir. Söz konusu değer ve ayrıca diğer performans ölçüm değerleri ele alındığında, Gini algoritmasına göre daha üstün performans gösterdiği ortaya çıkan C5.0 algoritmasının sonuçları karar kuralları elde etmek üzere değerlendirmeye alınmıştır.

Karışıklık matrisinde yer alan diğer performans ölçüm değerlerinin anlamı ve hesaplama yöntemlerini ifade etmek için bu konuya makalenin sonunda ayrıca yer verilecektir.

```
> cm_c50
Confusion Matrix and Statistics

          Reference
Prediction High Low Middle Very Low
  High       24   0      0        0
  Low         0  31      1        1
  Middle      1   1     29        0
  Very Low    0   0      0       11

Overall Statistics

               Accuracy : 0.9596
                 95% CI : (0.8998, 0.9889)
    No Information Rate : 0.3232
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9441
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: High Class: Low Class: Middle Class: Very Low
Sensitivity               0.9600     0.9688        0.9667          0.9167
Specificity               1.0000     0.9701        0.9710          1.0000
Pos Pred Value            1.0000     0.9394        0.9355          1.0000
Neg Pred Value            0.9867     0.9848        0.9853          0.9886
Prevalence                0.2525     0.3232        0.3030          0.1212
Detection Rate            0.2424     0.3131        0.2929          0.1111
Detection Prevalence      0.2424     0.3333        0.3131          0.1111
Balanced Accuracy         0.9800     0.9694        0.9688          0.9583
```

Şekil 17: C5.0 Algoritması ile İlgili Karışıklık Matrisi ve Performans Değerleri

C5.0 algoritması ile elde edilen karar kurallarını listelemek için s*ummary()* fonksiyonuna başvurulur.

```
C50 <-C5.0(UNS ~., data=egitim)
summary(C50)
```

Şekil 18: C5.0 Algoritmasının Karar Kuralları ile İlgili Komut Dizilimi

Bu fonksiyon aracılığıyla elde edilmiş olan karar kuralları Şekil 19'da görülmektedir.

```
> summary(C50)
Call:
C5.0.formula(formula = UNS ~ ., data = egitim)

C5.0 [Release 2.07 GPL Edition]       Mon Nov 13 15:07:24 2017
-------------------------------

Class specified by attribute `outcome'

Read 304 cases (6 attributes) from undefined.data

Decision tree:

PEG > 0.4:
:...PEG > 0.68: High (69/1)
:   PEG <= 0.68:
:   :...LPR <= 0.8: Middle (75/1)
:       LPR > 0.8: High (8)
PEG <= 0.4:
:...PEG <= 0.15:
    :...LPR <= 0.57: Very Low (32)
    :   LPR > 0.57: Low (13/2)
    PEG > 0.15:
    :...LPR <= 0.6:
        :...LPR <= 0.03: Very Low (2)
        :   LPR > 0.03: Low (68/2)
        LPR > 0.6:
        :...PEG > 0.27: Middle (12)
            PEG <= 0.27:
            :...PEG <= 0.23: Low (11)
                PEG > 0.23:
                :...STR <= 0.35: Low (6)
                    STR > 0.35:
                    :...SCG <= 0.31: Low (4/1)
                        SCG > 0.31: Middle (4)
```

Şekil 19. C5.0 Algoritması ile İlgili Karar Kuralları

2.2.3.3. Sınıflandırma Algoritması ile Elde Edilen Karar Kurallarının İfadesi

Karar ağacı oluşturan sınıflandırma algoritmaları ile elde edilen karar kurallarının incelenmesi sonucunda her biri, aşağıda olduğu gibi ifade edilebilir. Bunun için, her düğüme ait bir kural oluşacak şekilde numara verilir ve *"Eğer.... ve ..."* yapısında bir ifadeye dönüştürülür. C5.0 algoritması ile geliştirilen karar ağacı için oluşturulan karar kurallarının ifadesi Şekil 20'de görülmektedir.

Kural1: Eğer PEG > 0.4.
ve PEG > 0.68 ise USD = "High"

Kural2: Eğer PEG > 0.4.
ve PEG <= 0.68.
ve LPR <= 0.8 ise USD = "Middle"

Kural3: Eğer PEG > 0.4.
ve PEG <= 0.68.
ve LPR > 0.8 ise USD = "High"

Kural4: Eğer PEG <= 0.4.
ve PEG <= 0.15.
ve LPR <=0.57 ise USD = "Very Low"

Kural5: Eğer PEG <= 0.4.
ve PEG <= 0.15.
ve LPR > 0.57 ise USD = "Low"

Kural6: Eğer PEG <= 0.4.
ve PEG > 0.15.
ve LPR <= 0.6.
ve LPR <= 0.03 ise USD = "Very Low"

Kural7: Eğer PEG <= 0.4.
ve PEG > 0.15.
ve LPR <= 0.6.
ve LPR > 0.03 ise USD = "Low"

Kural8: Eğer PEG <= 0.4.
ve PEG > 0.15.
ve LPR > 0.6.
ve PEG > 0.27 ise USD = "Middle"

Kural9: Eğer PEG <= 0.4
ve PEG > 0.15
ve LPR > 0.6
ve PEG <= 0.27
ve PEG <= 0.23 ise USD = "Low"

Kural10: Eğer PEG <= 0.4.
ve PEG > 0.15.
ve LPR > 0.6.
ve PEG <= 0.27.
ve PEG > 0.23.
ve STR <= 0.35 ise USD = "Low"

Kural11: Eğer PEG <= 0.4.
ve PEG > 0.15.
ve LPR > 0.6.
ve PEG <= 0.27.
ve PEG > 0.23.
ve STR > 0.35.
ve SCG <= 0.31 ise USD = "Low"

Kural12: Eğer PEG <= 0.4.
ve PEG > 0.15.
ve LPR > 0.6.
ve PEG <= 0.27.
ve PEG > 0.23.
ve STR > 0.35.
ve SCG > 0.31 ise USD = "Middle"

Şekil 20: C5.0 Algoritması ile İlgili Karar Kurallarının İfadesi

2.2.3.4. Sınıflandırma Algoritması ile Elde Edilen Karar Kurallarının Yorumu

Kullanıcı modelleme sistemi aracılığıyla çevrim içi öğrenme ortamlarını kullanan öğrencilerin bilgi düzeylerinin sınıflandırılması amacıyla uygulanan C5.0 Algoritmasının sonuçları şu şekildedir:

Söz konusu algoritma ile elde edilen karar kuralları doğrultusunda; öğrencilerin hedef konular için yapılan sınavdaki performansları ile ilgili niteliğin (*PEG*), öğrencilerin bilgi seviyelerini en belirleyici nitelik olduğu ortaya çıkmıştır. Karar kurallarına göre, bilgi seviyesinin belirlenmesinde en önemli nitelik PEG iken, önem sırasında PEG'den sonra gelen niteliğin LPR olduğu görülmektedir. Buna göre; başta hedef konu için yapılan sınav olmak üzere, öğrencilerin her iki sınav türü için gerçekleşen performans seviyeleri bilgi düzeyini belirler durumdadır. Karar kuralları doğrultusunda, bir öğrencinin hedef konular için yapılan sınavdaki performansı $\frac{68}{100}$ 'den yüksek ise, başka niteliklerin derecelerine bakılmaksızın bu öğrencinin bilgi seviyesi yüksek ("*High*") olarak ele alınır.

Öğrencinin sınav performansı en az $\frac{40}{100}$ olmak kaydıyla, $\frac{68}{100}$'den düşük ise, öğrencinin hedef konu ile ilgili diğer konular için yapılan sınavlardaki performanslarına (LPR) bakılır. Bu sınavlardaki performansı $\frac{80}{100}$'in üzerindeyse, bilgi seviyesi yine "*yüksek*" olarak ele alınır. Eğer sınav performansı $\frac{80}{100}$'in altında ise bu seviye orta ("*Middle*") olarak değerlendirilir.

Karar kuralları incelendiğinde, PEG $\frac{40}{100}$'ın altına düşerse, bilgi seviyesinin en fazla *"orta"* olabileceği görülmektedir. Bunun için aşağıdaki koşulların gerçekleşmesi gerekir:

- PEG > $\frac{27}{100}$ ve LPR > $\frac{60}{100}$ şartlarının aynı anda gerçekleşmesi,
- PEG > $\frac{23}{100}$ olması şartıyla, eğer PEG <= $\frac{27}{100}$ ve LPR > $\frac{60}{100}$ ise, öğrencinin öğrenilecek bir konu ile ilgili olan diğer konuları çalışma süresinin (STR) ve öğrenilen materyalleri tekrar etme sayısının derecesine (SCG) bakılır. Bu değerler sırasıyla; STR > $\frac{35}{100}$ ve SCG > $\frac{31}{100}$ şartlarını sağlamalıdır.

Söz konusu iki koşul sağlandığı sürece, öğrencinin *"orta"* seviyede bilgi düzeyi olabileceği kabul edilir.

Elde edilen karar kuralları içindeki, mümkün diğer koşullarda bilgi düzeyinin seviyesi; düşük *"Low"* ya da çok düşük *"Very Low"* olacaktır.

Karar kurallarına göre, en kötü koşulu ortaya koyan niteliklerin yine PEG ve LPR değerleri olduğu görülmüştür. En kötü koşul aşağıdaki iki mümkün halde ortaya çıktığında, bilgi seviyesi "*çok düşük*" olarak ele alınır.

- PEG <= $\frac{15}{100}$ ve LPR <= $\frac{57}{100}$ olmalı.
- PEG > $\frac{15}{100}$ ve PEG <= $\frac{40}{100}$ ve ayrıca, LPR <= $\frac{3}{100}$ olmalı.

2.2.3.5. Sınıflandırma Algoritması ile Elde Edilen Karışıklık Matrisinin Yorumu

		Gerçek					
		High	**Low**	**Middle**	**Very Low**	**Toplam**	
Tahmin	**High**	24	0	0	0	**24**	
	Low	0	31	1	1	**33**	
	Middle	1	1	29	0	**31**	
	Very Low	0	0	0	11	**11**	
	Toplam	**25**	**32**	**30**	**12**	**N=99**	

Şekil 21: C5.0 Modeli İle Elde Edilen Karışıklık Matrisi

- **Accuracy**: Toplam veri içindeki doğru tahmin oranıdır (Markham, 2014).

$$Accuracy = \frac{Toplam\ doğru\ tahmin}{N} = \frac{24+31+29+11}{99} = 0.9596$$

- **95% CI: %95 aralıkla güven aralığı:** Model sonuçlarının doğruluğu (0.8998-0.9889) aralığında gerçekleşmektedir.
- **No Information Rate:** Pozitif olarak ele alınan sınıfın toplam tahmin içindeki oranıdır. Bu model için elde edilen elde edilen sonuçlar için "*Low*" sınıfı pozitif olarak ele alınmıştır.

$$No\ Information\ Rate = \frac{Toplam(gerçek(Low))}{N} = \frac{32}{99} = 0.3232$$

- **P_Value:** $2.\ 10^{-16}$
- **Kappa:** 0.9441
- **Mcnemar's Test P-Value: NA**

Aşağıda yer alan ölçeklerin modelde yer alan bütün sınıf değerleri için ayrı ayrı hesaplanması gerekir. Bu çalışmada pozitif sınıf olarak *"Low"* sınıfının ele alındığı varsayılarak her bir ölçeğin değeri hesaplanacaktır. Söz konusu ölçeklerin değerleri, verilen örnek çözümlere benzer yöntemle diğer sınıflar için de hesaplanabilir.

- **Sensitivity** (Doğru pozitif oran, DP)**:** Model sonucuna göre; bir sınıfın pozitif tahmin değerleri içinde doğru tahmin edilme olasılığını gösteren duyarlılık değeridir. Örneğin, gerçekte *"Low"* iken, modelin tahmin ettiği sınıfın da *"Low"* olma olasılığını ifade eder. Karışıklık tablosunda yer alan sütunlar, her sınıfa ait pozitif değerleri içermektedir.

  DP: Pozitif olarak belirlenen sınıfın kaç kere doğru olarak tahmin edildiğini gösteren değer.

$$Sensitivity_{Low} = \frac{DP}{Toplam(Gerçek_{Low})} = \frac{31}{32} = 0.9688$$

- **Specificity** (Doğru negatif oran, DN)**:** Model sonucuna göre; yapılan tahminler içinde, bir sınıfın gerçekleşmeme durumunun doğru olarak tahmin edilme olasılığını belirleyen bir değerdir. Örneğin, gerçekte *"Low"* değil iken, modelin tahmin ettiği sınıfın da *"Low"* olmama olasılığıdır.

Bu amaçla, öncelikle her sınıf için doğru negatif oran ve yanlış pozitif (YP) değerleri hesaplanır.

DN: Pozitif olarak ele alınan sınıfın (Low) satır ve sütunları dışında kalan matris değerlerinin (doğru negatiflerin) toplamı. (Low sınıfının tahminler içinde ortaya çıkmadığı durumlar)

YP: Low sınıfının satırındaki gerçekte Low olmayan bir sınıfın hatalı bir şekilde Low olarak tahmin edilme sıklığını gösteren değerler.

$$\text{Specificity}_{sınıf} = \frac{DN}{DN+YP}$$

$DN_{Low} = (24+0+0) + (1+29+0) + (0+0+11) = 65$

$YP_{Low} = 0+1+1=2$

$$\text{Specificity}_{Low} = \frac{65}{67} = 0.9701$$

- **Pos Pred Value** (Pozitif Tahmin Değeri)**:** Toplam pozitif tahminler içinde bir sınıfın doğru tahmin edilme olasılığıdır. Örneğin, model sonucuna göre tahmin edilen *"Low"* sınıfının gerçekte de *"Low"* olma olasılığını tanımlar.

$$\text{Pos Pred Value}_{sınıf} = \frac{DP}{DP+YP} = \frac{DP}{Toplam(pozitif)}$$

$DP_{Low} = 31$

$YP_{Low} = 0+1+1 = 2$

$$\text{Pos Pred Value}_{Low} = \frac{31}{33} = 0.9394$$

- **Neg Pred Value** (Negatif Tahmin Değeri)**:** Pozitif sınıf olarak belirlenen bir sınıfın dışındaki diğer sınıfların toplam negatif tahmin edilenlere oranıdır.

Bu amaçla, öncelikle her sınıf için doğru negatif oran (DN) ve yanlış negatif (YN) değerleri hesaplanır.

DN: Pozitif olarak ele alınan sınıfın satır ve sütunları dışında kalan doğru negatif değerlerin toplamı.

YN: Pozitif olarak ele alınan sınıfın sütununda yer alan ve doğru tahmin (DP) değerinin dışındaki yanlış negatif değerlerin toplamı.

$$\text{Neg Pred Value}_{sınıf} = \frac{DN}{DN+YN} = \frac{DN}{Toplam(negatif)}$$

$DN_{Low} = (24+0+0)+(1+29+0)+(0+0+11)=65$

$YN_{Low} = 0+1+0 =1$

$$\text{Neg Pred Value}_{Low} = \frac{65}{66} = 0.9848$$

Bu iki değer, aşağıda yer alan formüller aracılığıyla da hesaplanabilmektedir (Kuhn, 2017):

$$\text{Pos Pred Value} = \frac{(sensitivity * prevalence)}{(sensitivity * prevalence) + ((1 - specificity) * (1 - prevalence))}$$

- **Prevalence:** Toplam tahmin değerleri içinde bir sınıfın gerçek değerlerinin ortaya çıkma olasılığını veren bir yaygınlık ölçüsüdür. Bunun için, karışıklık matrisinde yer alan ve her sınıfa ait pozitif değerleri içeren sütunların alt toplamlarının toplam tahmin oranı ele alınır. (Kuhn, 2017).

$$\text{Prevalence}_{sınıf} = \frac{Toplam\,(Gerçek_{sınıf})}{N}$$

$$\text{Prevalence}_{Low} = \frac{32}{99} = 0.3232$$

- **Detection Rate:** Her bir sınıfın doğru tahmin değerinin toplam tahmin değerleri içindeki oranı.

$$\text{Detection Rate}_{sınıf} = \frac{DP}{N}$$

$$\text{Detection Rate}_{Low} = \frac{31}{99} = 0.3131$$

- **Detection Prevalence:** Her bir sınıfın toplam tahmin değerinin genel tahmin değerleri içindeki oranıdır. Bu değeri elde etmek için, pozitif olarak belirlenen sınıfın tahmin değerlerinin bulunduğu satırdaki, pozitif tahmin değeri (DP) ve yanlış pozitif değerleri (YP) toplamı ele alınır.

$$\text{Detection Prevalence}_{sınıf} = \frac{DP+YP}{N}$$

$$\text{Detection Prevalence}_{Low} = \frac{33}{99} = 0.3333$$

- **Balanced Accuracy:** Her sınıf için belirlenen dengelenmiş doğruluk değeridir.

$$\text{Balanced Accuracy}_{sınıf} = \frac{Sensitivity+Specificity}{2}$$

$$\text{Balanced Accuracy}_{Low} = \frac{0.9688+0.9701}{2} = 0.9694$$

3. SONUÇ ve ÖNERİLER

Bu çalışma kapsamında, öğrenmeye dayalı sınıflandırma yöntemlerinden C5.0 ve Gini algoritmaları ile bir veri madenciliği uygulaması gerçekleştirilmiştir. Bu amaçla açık kaynak kodlu R Dili ile çözüm sağlanmıştır. Uygulama sonunda, her iki algoritmanın oluşturduğu sınıflandırma modeli üzerinde *ConfusionMatrix()* fonksiyonu ile bir performans değerlendirmesi yapılmıştır. Gerçekleşen performans değerlendirmesi sonucu Gini algoritmasının C5.0 algoritmasına oranla daha düşük performans sahip olduğu görülmüştür. Bu nedenle, C5.0 algoritması ile oluşturulan sınıflandırma modeli ve modelin oluşturduğu karar ağaçları ve dolayısıyla karar kurallarından elde edilen sonuçlar değerlendirmeye alınmıştır.

C5.0 algoritması ile elde edilen karar kuralları aracılığıyla, veri setinde yer alan niteliklerin içinde, hedef nitelik üzerinde etkin olan niteliklerin varlığı önem sırasına göre ortaya konmuştur. Söz konusu karar kuralları doğrultusunda, hedef nitelik üzerinde en fazla etkili olan nitelikler önem sırasına göre aşağıda listelenmiştir:

1. Öğrencilerin hedef konular için yapılan sınavdaki performansı (*PEG*)

2. Öğrencilerin hedef konu ile ilgili diğer konular için yapılan sınavlardaki performansları (LPR)

Elde edilen sonuçlara göre, PEG niteliği öğrencilerin bilgi düzeyinin belirlenmesinde en önemli belirleyici nitelik olmuştur. Önem sırasında bu niteliği ikinci sıradaki LPR takip etmektedir.

Sınıflandırma algoritmalarının uygulanması aşamasında verinin analize uygun olması çok önemlidir. Aksi durumda, model çözümü hatalı bir şekilde sonlanacaktır. Veri içerisinde eksik veri ya da aykırı veri (outliers) bulunabilir. Böyle bir durumda, eksik verinin yerine ortalama ya da benzeri bir değer yerleştirilerek sorunun giderilmesi gerekir. Aykırı veri mevcut ise, Min-Max Normalizasyon ya da Zscore Standardizasyon gibi Normalizasyon tekniklerinden biri uygulanmalıdır. Bu teknikler aracılığıyla aykırı değerler eğer başka bir aralık tanımlanmazsa, 0-1 aralığında ölçeklenir.

Bu çalışmada sadece iki algoritma ile sınıflandırma modeli oluşturulmuştur. Verinin uygun olduğu bütün algoritmaların uygulanması ve performans değerlendirmesi doğrultusunda en başarılı performansa sahip modelin ele alınarak, çözüm gerçekleştirilmesi önerilir.

Yararlanılan Kaynaklar

Adak, M. F. & Yurtay, N. (2013). Gini Algoritmasını Kullanarak Karar Ağacı Oluşturmayı Sağlayan Bir Yazılımın Geliştirilmesi, *Bilişim Teknolojileri Dergisi*, Cilt: 6, Sayı: 3, 1-6.

Balaban, M. E. & Kartal, E. (2016). *Veri Madenciliği ve Makine Öğrenmesi Temel Algoritmaları ve R Dili ile Uygulamaları*, Çağlayan Yayınevi, İstanbul.

Cunningham, P., Cord, M. & Delany, S. J. (2008). Supervised Learning, Machine Learning Techniques for Multimedia, Chapter 2, *Springer*, 21-49.

Han, J. & Kamber, M. (2012). *Data Mining: Concepts and Techniques*, Elsevier Inc., Third Edition, USA.

Hastie, T., Tibshirani, R. & Friedman, J. (2008). *The Elements of Statictical Learning: Data Mining, Inference, and Prediction*, Second Edition, Springer.

Kahraman, H.T., Sağıroğlu, S. & Çolak, I. (2013). Developing Intuitive Knowledge Classifier And Modeling Of Users 'domain Dependent Data In Web, *Knowledge Based Systems*, vol. 37, 283-295.

Kantardzic, M. (2011). *Data Mining Concepts, Models, Methods, and Algorithms*, A John Wiley & Sons, Inc., Second Edition, USA.

Kohavi, R. & Provost, F. (1998). Glossary of Terms Special Issue on Applications of Machine Learning and the Knowledge Discovery Process, *Kluwer Academic Publishers*, Boston, http://robotics.stanford.edu/~ronnyk/glossary.html (05.11.2017).

Kuhn, M.(2017). *Classification and Regression Training, Package 'caret' Version 6.0-77*, https://cran.r-project.org/web/packages/caret/caret.pdf (05.11.2017).

Kuhn, M., Weston, S., Coulter, N. ,Culp, M. & Quinlan, R. (2018), *Decision Trees and Rule-Based Models, Package 'C50'*, https://cran.r-project.org/web/packages/C50/C50.pdf (05.10.2018).

Kumar, S. V. K. & Kiruthika, P. (2015). An Overview of Classification Algorithm in Data mining, *International Journal of Advanced Research in Computer and Communication Engineering-IJARCCE*, Vol. 4, Issue 12, 255-257, https://www.ijarcce.com/upload/2015/december-15/IJARCCE%2059.pdf (19.02.2018).

Markham, K. (2014). *Simple Guide To Confusion Matrix Terminology*, http://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/ (05.11.2017).

Özkan,Y. & Erol, Ç.S. (2015). *Biyoenformatik DNA Mikrodizi Veri Madenciliği*, Papatya Yayıncılık, İstanbul.

Özkan, Y. (2016). *Veri Madenciliği Yöntemleri*, Papatya Yayıncılık Eğitim A.Ş., Üçüncü Basım, İstanbul.

Pandya, R. & Pandya, J. (2015). C5.0 Algorithm to Improved Decision Tree with Feature Selection and Reduced Error Pruning, *International Journal of Computer Applications (0975 – 8887)*, Volume 117 – No. 16, 18-21, http://research.ijcaonline.org/volume117/number16/pxc3903318.pdf (19.02.2018).

Sokolova, M. & Lapalme, G.(2009). A Systematic Analysis of Performance Measures For Classification Tasks*, Information Processing and Management*, 45 (2009) 427–437, Elsevier Inc., http://atour.iro.umontreal.ca/rali/sites/default/files/publis/SokolovaLapalme-JIPM09.pdf (19.02.2018).

Therneau, T., Atkinson, B. & Ripley, B. (2017). *Recursive Partitioning and Regression Trees, Package 'rpart'*, https://cran.r-project.org/web/packages/rpart/rpart.pdf (05.11.2017).

UCI (2009). https://archive.ics.uci.edu/ml/datasets/User+Knowledge+Modeling (05.11.2017).

Ayşe ÇINAR – acinar@marmara.edu.tr

She works as an Assistant Professor at Marmara University, Faculty of Business Administration. She accomplished her master and Ph.D degrees in Quantitative Methods at Istanbul University, Institute of Social Sciences. She teaches Data Mining, Linear Programming, Decision Theory and Game Theory. Her research interests include data mining and operations management.