TİDE Academia Research
Sayı 1 / 71 -89 (2020)

Research Article/Araştırma Makalesi

BANKALARIN SAHTE PARA DENETİMİNDE KULLANDIKLARI KARAR AĞAÇLARINDA SONUÇ İYİLEŞTİRİCİ YÖNTEMLER

Batuhan BİLENLER[1]
Kutluk ÖZGÜVEN[2]

Submitted/Başvuru: 21.10.2019
Revised/Düzeltme:15.05.2020
Accepted/Kabul: 28.06.2020

Öz

Bu çalışmada, sahte para denetimi konusunda çevrimiçi platform üzerinden alınan hazır verilerin Güdümlü Makine Öğrenmede kullanılan Karar Ağacı algoritmalarından CART algoritması ile sınıflandırma işlemi yapılırken tüm süreçlerinin incelenmesi ve algoritmik iyileştirmeler yapılması amaçlanmaktadır. Çapraz doğrulama aşamasında eğitim verilerinin daha doğru seçilmesi yolu ile ağaç yapısının daha doğru eğitilmesi, bu şekilde sahtecilik başta olmak üzere algoritmanın her tür denetimde daha yüksek etkinlikte kullanılması beklenmektedir. Test ve eğitim verileri alt kümelere ayrılırken belirli kriterlere göre bu işlemlerin yapılması özellikle eğitim aşamasında sistemin kararlılığını artırmakta ve başarı oranını yükseltmektedir. Veri setini, alt veri setlere ayırma aşamasında sınıflandırma sonucuna etkisini artırmak amacıyla algoritmik geliştirmeler yapılmıştır.

Anahtar Kelimeler: Sahte Para Denetimi, Karar Ağaçları, CART Algoritması, Makine Öğrenmesi, Karar Destek Sistemleri

JEL Sınıflandırması: C87, M42, Y10

1 Sorumlu Yazar, Yüksek Lisans Öğrencisi, İstanbul Aydın Üniversitesi Bilgisayar Mühendisliği, bilenlerbatuhan@gmail.com , ORCID ID: 0000 0002 7295 4876.

2 Profesör Doktor, Tez Danışmanı, İstanbul Aydın Üniversitesi Bilgisayar Mühendisliği, kutlukozguven@aydın.edu.tr, ORCID ID: 0000-0001-6372-6640.

To cite this article: Bilenler, B. & Özgüven K. (2020). Bankaların Sahte Para Denetiminde Kullandıkları Karar Ağaçlarında Sonuç İyileştirici Yöntemler. *TİDE AcademIA Research*, 2(1), 71-89.

TİDE Academia *Research*
Sayı 1 / 71 -89 (2020)

METHODS TO INCREASE EFFECTIVENESS OF DECISION TREES IN COUNTERFEIT DETECTION IN BANKING

Abstract

In this study it is aimed to examine counterfeiting detection processes data gathered from an established online platform through application of the supervised machine learning decision tree CART algorithm and to make algorithmic improvements in order to increase the effectiveness of the algorithm on similar detection and banking data analysis and learning requirements. In Cross Validasyon phase, it is expected that the tree structure will be trained more accurately by preselecting training data more accurately. When testing and training data is subdivided, performing these procedures according to certain criteria increases the stability of the system and the success rate, especially during the training phase. Algorithmic improvements are applied in order to increase the effect of the data to be used in the division of data set n sub-section on the classification result.

Keywords: Counterfeit Detection, Decision Trees, CART Algorithm, Machine Learning, Decision Support Systems

JEL Classification: C87, M42, Y10

Extended Summary

In the decision tree classifier, as a result of the algorithmic improvements made to divide the data into sub-datasets at the cross validation stage, it is seen that there is a more accurate tree structure and an increase in performance in the max score and mean scores. When it is desired to make a classification process in the industrial area, the importance of determining the properties with the least and the most impact on the classification has been revealed by conducting a preliminary study. After the new algorithmic developments, when float data type is used, it is foreseen that the accuracy rates will increase considerably especially in the financial sector such as customer risk analysis and fraud detection. It is thought that the success to be achieved in the industrial field will increase the interest in classification algorithms and R&D studies in the coming years.

Introduction

In this study, it was aimed to classify the data correctly by making algorithmic developments on the CART algorithm created using the decision tree. The decision tree technique, which can be used for any decision problem, is particularly useful in the display of decision problems that require multiple sequential decisions. Today, many techniques are used for classifying data. Classification is a predictive model and it is a classification process to predict how the weather will be the next day or how many blue balls are in a box. Decision Trees, Logistic Regression, Artificial Neural Networks, kNN, Bayes and Fuzzy Logic are the algorithmic artificial intelligence and machine learning methods.

If the dependent variable under consideration is categorical, the method is called Classification Tree, whereas the regression tree is called Regression Tree. Decision trees are a structure used to divide a dataset containing a large number of records into very small groups of records, using simple decision making steps. What is the algorithm used when creating decision trees is an important consideration.

Literature Review

It creates an index value between 0 and len (dataset) using the randrange() function in the Random class, while dividing the data into sections using the n_folds parameter in

the current CART algorithm. The element with this index value is placed in folds. n-1 subset training and 1 test data, sub-data sets are created. Then, control is performed by cross-verifying for each subset. The Cross Validation method makes choices with the smallest possible error. The purpose of separating the data set into training and test sets is to avoid possible overfitting and to understand how the model performs on the data set that it has not seen before.

Methodology

In this study, instead of an index produced between 0 and len (dataset) using the randrange () function in the Random class in the current CART algorithm, an index value will be generated by operating the following steps.

Step 1: Identify the column with the highest impact on the classification result.

Step 2: Identify the column with the lowest effect on the classification result.

Step 3: Calculate the average value of the column with the highest effect on the classification result.

Step 4: Calculate the average value of the column with the lowest effect on the classification result.

Step 5: If the average value of the column with the highest effect on the classification result in the loop structure is larger than the next value while circulating the data on the same column, keep the index and exit the loop.

Step 6: If the average value of the column with the lowest effect on the classification result in the loop structure is greater than the next value while navigating the data on the same column, keep the index and exit the loop.

Step 7: If Step 5 and Step 6 do not provide both, generate an index value between 0 and len (dataset) using the randrange () function inside the Random class and exit the loop

Conclusion

As a result of the algorithmic improvements made to divide the data into 5 sub-sets during

the Cross Validation stage, it is seen that there is a more accurate tree structure and an increase in performance in the max score and mean scores. When it is desired to make a classification process in the industrial area, the importance of determining the properties with the least and the most impact on the classification has been revealed by conducting a preliminary study. After the new algorithmic developments, when float data type is used, it is foreseen that the accuracy rates will increase considerably especially in the financial sector such as customer risk analysis and fraud detection. It is thought that the success to be achieved in the industrial field will increase the interest in classification algorithms and R&D studies in the coming years.

1. Giriş

Bu çalışmada Karar Ağacı kullanarak oluşturulan CART algoritması üzerinde algoritmik geliştirmeler yaparak verileri doğru şekilde sınıflandırmak amaçlanmıştır. Herhangi bir karar problemi için kullanılabilen Karar Ağacı tekniği özellikle birden fazla kararın ardışık olarak verilmesini gerektiren karar problemlerinin gösteriminde çok kullanışlıdır (Albright, Winston & Zappe, 2006). Günümüzde verileri sınıflandırma işlemi için birçok teknik kullanılmaktadır. Sınıflama tahmin edici bir model olup, havanın bir sonraki gün nasıl olacağı veya bir kutuda kaç tane mavi top olduğunun tahmin edilmesi bir sınıflama işlemidir (Silahtaroğlu, 2008). Bu metodlardan en çok kullanılanları Karar Ağaçları, Lojistik Regresyon, Yapay Sinir Ağları, kNN, Bayes ve Bulanık Mantık algoritmik yapay zeka ve makine öğremesi yöntemleridir.

Ele alınan bağımlı değişken kategorik ise yöntem sınıflama ağaçları (Classification Tree), sürekli ise regresyon ağaçları (Regression Tree) olarak adlandırılmaktadır (Deconinck, 2005). Karar ağaçları, basit karar verme adımları uygulanarak, çok sayıda kayıt içeren bir veri kümesini çok küçük kayıt gruplarına bölmek için kullanılan bir yapıdır (Berry & Linoff, 2004). Karar ağaçları oluşturulurken kullanılan algoritmanın ne olduğu önemli bir husustur. Burada, amaç-hedef değişkene ilişkin mümkün olabilen en homojen veri alt gruplarını üretmektir.(Kurt & Ture 2008). Kullanılan algoritmaya göre ağacın şekli değişebilir. Bu durumda değişik ağaç yapıları da farklı sınıflandırma sonuçları verecektir. Kök denilen ilk düğümün farklı olması, en uçtaki yaprağa ulaşırken izlenecek yolu ve dolayısıyla sınıflandırmayı da değiştirecektir (Silahtaroğlu, 2008). CART'ın sahip olduğu algoritma, benzerlik gösteren değişkenlerin aynı ağaç düğümünde toplanmasına dayalı olup, bütün oluşturduğu alt dalları bağımlı değişken olan kök düğüme bağlamayla son bulmaktadır (Teng, Lin & Ho, 2007). Her düğüm sürekli ikiye bölünür. Bölünecek noktaların belirlenmesinde sıklıkla Gini veya Twoing gibi ayırma ölçütleri kullanılmaktadır.

CART algoritmasında bir düğümde belirli bir kriter uygulanarak bölünme işlemi gerçekleştirilir. Her düğümden ancak 2 alt dal ayrılabilir. CART algoritması CHAID algoritmasının daha gelişmiş hali olarak yorumlanabilir. CHAID algoritması çoklu kırılım modelini benimsemektedir. Bu özelliğinden dolayı ticari faaliyetlerde kullanımı daha yaygındır. CART algoritması ise sadece ikili kırılıma izin verdiği için yüksek kestirim yapılması iste-

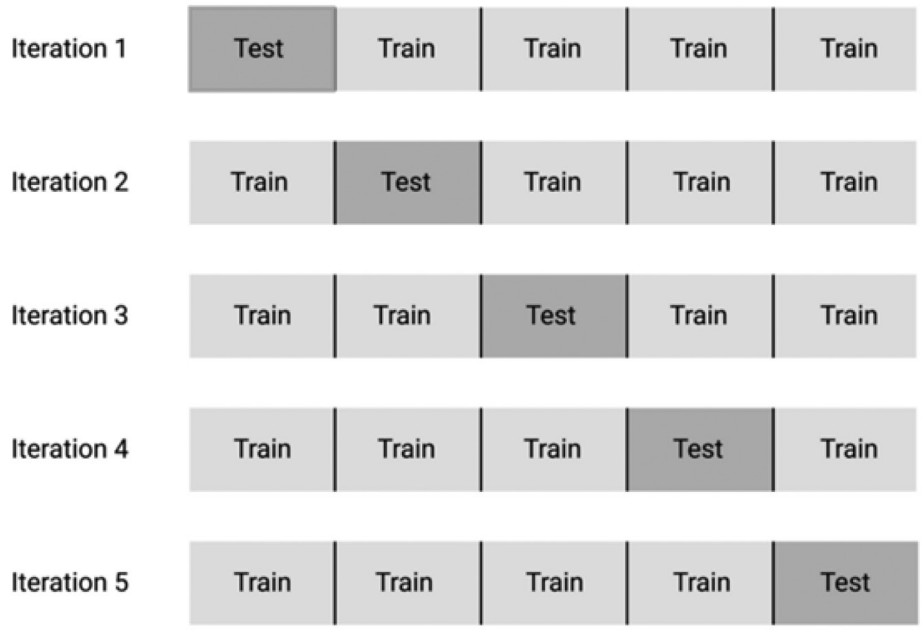
nilen alanlarda daha çok tercih edilmektedir. Değişken adetinin veya kategorisinin fazla olduğu durumda ikili ağaç yapısı daha iyi sonuçlar üretebilir (Lewis, 2000).

CART algoritması, (a) kirli verilerle çalışmaya izin verebilir, (b) Gini index değerine göre kırılımları belirler, (c) Sayısal değerleri input olarak çalışmaya izin verir, (d) hem sınıflandırma hem de regresyon tahminine olanak tanır, (e) eksik veriler ile çalışabilme özelliğine sahiptir, (f) 'ağaç budama' işlemini destekler.

CART algoritması, genellikle ağaç oluşturmak için kullanılan bir algoritmadır. Ağaç yapısının daha doğru ve dengeli olması hedeflenmiştir. Karar Ağacı üzerinde düğümleri birbirine bağlayan çizgilere dal adı verilir (Gordon & Pressman, 1983). İkili ağaç yapısını kullanır. . Bölümlendirme işlemi çocuk düğümlere (child node) veya alt düğümlerin her birine ardışık olarak uygulanır (Hand, Manila & Smyth, 2001). Bu yüzden sınıflandırma problemlerinin çözümünde önemli kolaylıklar sağlar. CART, sayısal ve nominal değerler üzerinde çalışabilir (Bozan, 2010). CART algoritması ile sınıflandırma yapılırken 3 önemli aşama bulunmaktadır. Bunlar, ağaç derinliğini belirleme, ağaç sayısını ve verilerin kaç parçaya ayrılacağını belirleme ve ayrılmış test verilerini ağaca uygulamadır.

2. Literatür Taraması

Mevcut CART algoritmasında n_folds parametresi kullanılarak veriseti bölümlere ayrırken Random sınıfının içinde bulunan randrange() fonksiyonu kullanılarak 0 ile len(dataset) arasında bir index değer oluşturur. Bu index degere sahip eleman fold'lara yerleştirilir. n-1 adet alt küme eğitim ve 1 adet test verisi olmak üzere alt veri setleri oluşturulur. Daha sonra her bir alt küme için çapraz doğrulama yapılarak kontrol gerçekleştirilir. Çapraz Validasyon (Cross Validation) yöntemi mümkün olan en küçük hata ile seçimleri yapar. Arlot ve Celisse (2010) Şekil 1'de verilerin alt veri setlerine ayrılması ve çapraz doğrulamanın yapılışı gösterilmektedir. Veri setini eğitim ve test set olarak ayrılmasının amacı, olası aşırı uyumdan (overfitting) kaçınmak ve modelin daha önceden görmediği veri seti üzerinde nasıl performans gösterdiğini anlamak içindir.



Şekil 1: Tüm Veri Setinden 5 Alt Veri Setine Ayrılarak Çapraz Doğrulama İşleminin Yapılması, Shaikh (2018).

3. Metodoloji ve Veriler

Bu bölümde araştırmada hangi yöntemleri kullanacağımız, hangi algoritmik değişikliklerle iyileştirme elde edileceği ve kullanılacak veri setini ele alınacaktır.

3.1. Çapraz Validasyon Aşamasında Algoritmik Geliştirmeler

Çapraz Validasyon aşamasında tüm veri setinden n adet alt veri setine verileri yerleştirirken belirli kriterlere göre ayrılıp, bazı alt kümelerde çok daha yüksek oranla başarı sağlanırken bazı alt kümelerde ise başarı oranında düşme meydana gelecektir. Mevcut algoritma yapısında n adet alt veri seti doğruluk oranları ayrı ayrı hesaplanmaktadır. N adet verinin aritmetik ortalaması bize ortalama doğruluk oranını vermektedir. Yapılacak olan algoritmik değişiklik ile birlikte performans artışı sağlanan verilerdeki doğruluk oranındaki

yükselme, performans kaybı yaratacak alt veri seti doğruluk oranlarından daha yüksek olacağı için ortalama değer daha da yükselecektir. Böylece daha doğru sınıflandırma yapılmış olacaktır.

Bu çalışmada, mevcut CART algoritmasında Random sınıfının içinde bulunan randrange() fonksiyonu kullanılarak 0 ile len(dataset) arasında üretilen bir index yerine, aşağıda belirtilen aşamalar işletilerek bir index değer üretilecektir.

1. Adım:Sınıflandırma sonucuna etkisi en yüksek sütunu belirle.
2. Adım: Sınıflandırma sonucuna etkisi en düşük sütunu belirle.
3. Adım: Sınıflandırma sonucuna etkisi en yüksek sütunun ortalama değerini hesapla.
4. Adım: Sınıflandırma sonucuna etkisi en düşük sütunun ortalama değerini hesapla.
5. Adım: Döngü yapısı içerisinde sınıflandırma sonucuna etkisi en yüksek sütunun ortalama değeri, aynı sütun üzerindeki verileri dolaşırken sıradaki değerden büyükse indexi tut ve döngüden çık.
6. Adım: Döngü yapısı içerisinde sınıflandırma sonucuna etkisi en düşük sütunun ortalama değeri, aynı sütun üzerindeki verileri dolaşırken sıradaki değerden büyükse indexi tut ve döngüden çık.
7. Adım: Adım 5 ve Adım 6 aşamalarının ikisini de sağlamıyorsa, Random sınıfının içinde bulunan randrange() fonksiyonu kullanılarak 0 ile len(dataset) arasında bir index değer üret ve döngüden çık.

3.2. Algoritmik Ön Çalışmaların Gerçekleştirilmesi

Adım 1 ve Adım 2 aşamalarında 4 input değerine sahip bir veri setinin sonuca en çok ve en az etkisi olan sütunları belirlemek için ayrı çalışma yapılacaktır. Bu ikinci çalışma için Spyder platformu üzerinde Python 3.7 sürümü kullanılacaktır. Bu python projesi içerisine Pandas ve Scikit-learn kütüphaneleri eklenmiştir. Python yazılım dili bu kütüphaneler sayesinde birçok işlemi kısa ve kolay şekilde yapmaya imkan vermektedir. Pandas kütüphanesi dosya okuma, satır ve sütunlar üzerinde ön işleme yapma, veriler üzerinde istatiksel bilgiler ortaya çıkarma gibi konularda pratik ve kullanıcı dostu bir kütüphanedir.

dataset2 - DataFrame

Index	3.6216	8.6661	-2.8073	-0.44699	0
0	4.5459	8.1674	-2.4586	-1.4621	0
1	3.866	-2.6383	1.9242	0.10645	0
2	3.4566	9.5228	-4.0112	-3.5944	0
3	0.32924	-4.4552	4.5718	-0.9888	0
4	4.3684	9.6718	-3.9606	-3.1625	0
5	3.5912	3.0129	0.72888	0.56421	0
6	2.0922	-6.81	8.4636	-0.60216	0
7	3.2032	5.7588	-0.75345	-0.61251	0
8	1.5356	9.1772	-2.2718	-0.73535	0
9	1.2247	8.7779	-2.2135	-0.80647	0
10	3.9899	-2.7066	2.3946	0.86291	0
11	1.8993	7.6625	0.15394	-3.1108	0
12	-1.5768	10.843	2.5462	-2.9362	0
13	3.404	8.7261	-2.9915	-0.57242	0

Şekil 2: Pandas Kütüphanesi ile Verilerin Okunması

Şekil 2'de Pandas kütüphanesi kullanılarak verilerin python projesi üzerinden okunması gösterilmiştir. Sırasıyla 0,1,2,3 nolu sütunlar sisteme giriş verisi olarak verilecektir. 4 no'lu index ise sınıflandırmanın sonucu olan çıkış verisini verecektir. Scikit-learn kütüphanesi içerisinde yer alan train_test_split nesnesi yardımıyla test ve eğitim verileri otomatik şekilde ayrılacaktır. Bu ayrım sırasında en dikkat edilmesi gereken test_size parametresidir. Parametrik değer ataması istenildiği oranda yapılabilinir; fakat genellikle kullanılan %25'i test ve %75'i eğitim şeklindedir. Herhangi bir oran belirtilmeden train_test_split(X,y) şeklinde bırakılırsa, default değer olarak arka planda 0.25 olarak ayrım gerçekleşecektir. Şekil 3.'te 1. Sütunun giriş parametresi olarak belirlenmesi ve train_test_split nesnesinin kullanımı gösterilmiştir.

```
@author: Batuhan
"""

import pandas as pd
from sklearn.model_selection import train_test_split

# csv dosyamızı okuduk.
dataset2 = pd.read_csv('data_banknote_authentication.csv.csv')

X = dataset2.iloc[:, [0]].values
y = dataset2.iloc[:, 4].values

x_train, x_test, y_train, y_test = train_test_split(X,y,test_size=0.25,random_state=0)
```

Şekil 3: Sütunun Giriş Parametresi Olarak Gösterilmesi

Scikit-learn kütüphanesi sınıflandırma işlemini ID3 algoritmasını baz alarak yapmaktadır. Karar ağacı sınıflandırması için, DecisionTreeClassifier sınıflandırıcı nesnesi kullanılacaktır.

DecisionTreeClassifer nesnesi birçok parametre ile kullanıcıdan istenilen özellikler doğrultusunda bir sınıflandırma yapmaya imkan vermektedir. En çok kullanılan parametreler:

criterion: Bölünmenin kalitesini belirlemek için kullanılır. DecisionTreeClassifer'ın desteklediği kriterler: gini safsızlığı için gini, bilgi kazanımı için ise entropi'dir. Opsiyonel bir parametredir. Belirtilmediği durumlarda arka planda default değer olarak gini kriteri üzerinden ayrım işlemlerini yapmaktadır.

splitter: Her düğümdeki bölünmeyi belirlemek için kullanılacak strateji parametresidir. Opsiyoneldir. best veya random olmak üzere 2 farklı yöntemden biri seçilebilinir. Belirtilmediği durumlarda ise arka planda best parametresi üzerinden strateji belirlenecektir.

max_depth: Bu parametre ağacın maximum gidebileceği derinliği belirlemek için kullanılır. Veri türü olarak integer değer almak zorundadır. Opsiyonel bir parametredir. Ağacın maksimum derinliği belirtilmemiş ise, tüm yapraklar saf olana kadar veya tüm yapraklar min_samples_split örneklerinden daha az içerinceye kadar düğümler genişletilir.

min_samples_split : Bir iç düğümünü bölmek için minimum örnek sayı bilgisini tutar. Arka planda parametre değeri belirtilmediği durumlarda 2 üzerinden işlemleri yapar.

Bu çalışmada, DecisionTreeClassifier nesnesi için criterion: 'gini' ve max_depth:10 başlangıç değerleri atanarak sınıflandırma yapılacaktır. Şekil 4.'te parametre atamalarının yapılması gösterilmiştir.

```
19 # DecisionTreeClassifier sınıfını import ettik
20 from sklearn.tree import DecisionTreeClassifier
21
22 # DecisionTreeClassifier sınıfından bir nesne ürettik
23 dtc = DecisionTreeClassifier(random_state=0,criterion='gini',max_depth=
```

Şekil 4: Sınıflandırıcının Projeye Eklenmesi ve Parametre Atamalarının Yapılması

x_train ve y_train eğitim verileri fit() fonksiyonu yardımıyla ağaç eğitilmektedir. x_test verileri ise predict() fonksiyonu kullanılarak tahminleme işlemi gerçekleştirilecektir. Bu fonksiyonlar scikit-learn kütüphanesinin kullanıcılara sağladığı büyük bir kolaylıktır. Şekil 5'te ağacın eğitilmesi ve tahminde bulunulması işlemleri gösterilmiştir. Tahmin edilen değerler ile gerçek değerlerin ne kadarının gerçekleştiğini ise accuracy_score fonksiyonu yardımıyla hesaplanmaktadır.

```
25 # Makineyi eğitiyoruz
26 dtc.fit(x_train,y_train)
27
28 # Test veri kümemizi verdik ve tahmin işlemini gerçekleştirdik
29 result = dtc.predict(x_test)
30
```

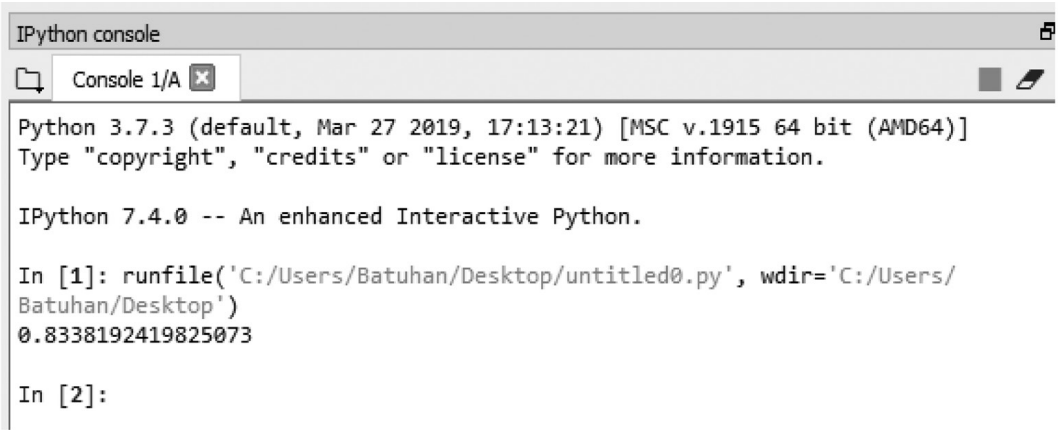
Şekil 5: Eğitim ve Tahminleme İşlemlerinin Uygulanması

Şekil 6'da makinanın tahmin ettiği değerler ile gerçek değerlerin yüzdesel olarak ne kadarının doğru çıktığını öğrenmek için accuracy_score nesnesinin kullanımı gösterilmektedir.

```
31 # Başarı Oranı
32 from sklearn.metrics import accuracy_score
33 accuracy2 = accuracy_score(y_test, result)
34
35 print(accuracy2)
```

Şekil 6: Doğruluk Oranının Hesaplanmasının Gösterimi

0 nolu index için sınıflandırma sonucuna tek başına etkisini bulmak için algoritma çalıştığında, Şekil 7'de görüldüğü gibi yüzdesel bir oran elde edilmiştir. Bu sonuç diğer sütunların hiçbir katkısı olmadan elde edilmiş bir değerdir.



Şekil 7: Algoritmanın Sadece 0 No'lu İndex için Çalıştığında Sınıflandırmaya Olan Etkisi

4 giriş sütunu için bu işlemler ayrı ayrı yapılıp, sütunların tek başına sınıflandırma sonucuna etkisi görülecektir. Bu işlemler sonucunda sınıflandırma için en önemli ve en az etkiye sahip sütunlar belirlenecek ve hipotezin Adım 1 ve Adım 2 aşamalarının yanıtları bulunmuş olacaktır. 4 sütun için de bu çalışma yapıldığında elde edilen sonuçlar Tablo 1.'de gösterilmiştir. Sonuçlar noktadan sonra 2 basamak olacak şekilde düzenlenmiştir. Bu yapılan ek çalışma sonucunda en önemli sütun 1 no'lu, en az etkiye sahip sütun ise 4 no'lu sütun olduğu sonucu ortaya çıkmıştır. Bu sütun index değerleri hipotezin 3.aşamasına geçmek için kullanılacaktır.

Tablo 1: Giriş Sütunlarının Sınıflandırma Sonucuna Etkisi

Sütun No	Tek Başına Sınıflandırma Sonucuna Etkisi
1.Sütun	%83.38
2.Sütun	%70.26
3.Sütun	%65.88
4.Sütun	%53.06

4. Uygulama

Metodoloji kısmında bahsedilen yöntemlerin uygulanması açıklandığı uyarınca gerçekleştirilmiş ve sonuçlar elde edilmiştir.

4.1 Ön Çalışmada Elde Edilen Verilerin Algoritma İyileştirmede Kullanılması

Adım 1 ve Adım 2 aşamasında görüleceği üzere sınıflandırma sonucuna tek başına en yüksek etkiyi yapan 1.sütun, en az etkiyi yapan ise 4. sütundur. Adım 3 ve Adım 4 aşamalarında ise float değerlere sahip bu sütunların ortalama değerleri hesaplanacaktır. Çapraz Validasyon aşamasında, sırayla veriler ortalama değerden büyük olmasına göre karşılaştırılaştırıp ona alt veri setlerine yerleştirilecektir. Aşağıda Şekil 8'de 2 sütun için ortalama değerleri hesaplayan fonksiyonlar gösterilmektedir.

```
def best_column_find_Average(dataset_copy):
    total=0
    for i in range(len(dataset_copy)):
         total=total+float(dataset_copy[i][0])

    return float(total/float(len(dataset_copy)))

def worst_column_find_Average(dataset_copy):
    total=0
    for i in range(len(dataset_copy)):
         total=total+float(dataset_copy[i][3])

    return float(total/float(len(dataset_copy)))
```

Şekil 8: Sınıflandırmaya Etkisi En Fazla ve En Az Olan Sütunların Ortalama Değerinin Bulunması

Çapraz Validasyon işleminde alt veri setleri oluşturulurken hangi indeksli verinin yerleşeceğine karar vermek için aşağıda belirtilen algoritma kullanılmıştır. Sırayla ana veri setindeki veriler alınıp, get_best_index isimli fonksiyon öncelikle ağırlığı en yüksek olan sütunun ortalama değerinde büyük olma durum kontrolü yapılacaktır. Bu koşulu sağlamıyorsa ağırlığı en az olan sütunun ortalama değerinden büyük olma durum kontrolü yapılacaktır. Bu iki koşulu da eğer sağlamıyorsa, Random sınıfının içerisinde bulunan randrange() fonksiyonu ile 0 ile len(dataset) arasında bir değeri üretip, return edecektir. Fonksiyonun algoritması Şekil 9'da gösterilmektedir.

```
def get_best_index(dataset_copy):
    for i in range(len(dataset_copy)):

        if (float(dataset_copy[i][0])>best_column_find_Average(dataset_copy)):
             break
        if (float(dataset_copy[i][3])>worst_column_find_Average(dataset_copy)):
             break
        else:
             return randrange(len(dataset_copy))

    return i

# Split a dataset into k folds
def cross_validation_split(dataset, n_folds):
    dataset_split = list()
    dataset_copy = list(dataset)
    fold_size = (int(len(dataset) / n_folds))

    for i in range(n_folds):
        fold = list()
        while len(fold) < fold_size:

            index =get_best_index(dataset_copy)
            fold.append(dataset_copy.pop(index))
        dataset_split.append(fold)
    return dataset_split
```

Şekil 9: Çapraz Validasyon Aşamasının Algoritmasının Gösterimi

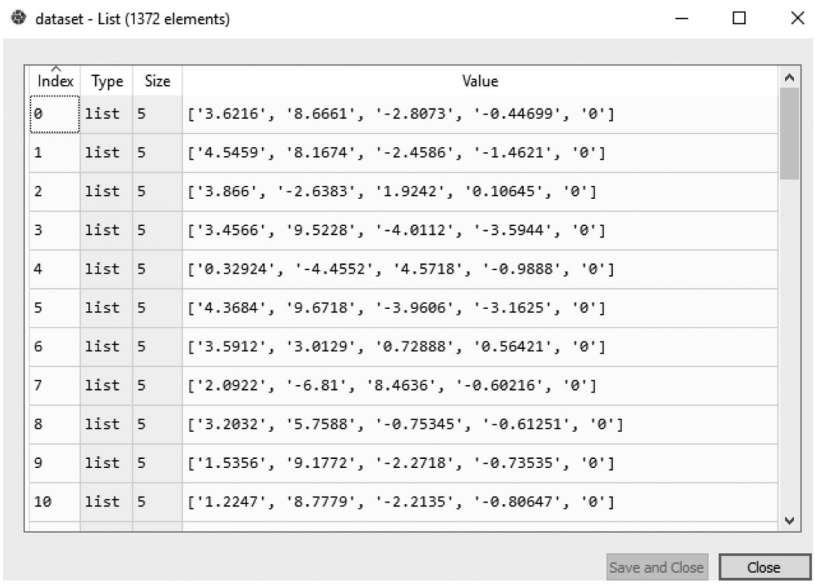
4.2. Uygulamanın Yapıldığı Veri Seti

Sınıflandırma çalışması için Kaggle platformu üzerinden tedarik edilen data_banknote_authentication isimli veri seti kullanılmaktadır. Bu veri seti 4 input ve 1 output değeri içermektedir. orjinal ve sahte banknot benzerlerinden alınmış örnek görüntülerden faydalanılarak hazırlanmıştır. Dijitalleştirme işlemi için endüstriyel kamera ile 400*400 pixel görüntüler kullanılmıştır. Yaklaşık 660 dpi çözünürlükte gri renkli fotoğraflardan özellikler çıkarabilmek için dalgacık dönüşümü aracı kullanılarak elde edilmiştir. Tablo 2'de veri setinde yer alan sütunlar ile ilgili bilgiler gösterilmektedir.

Tablo 2: Veri Setindeki Sütun İsimleri

Input 1	Input 2	Input 3	Input 4	Output
Dalgacık Dönüştürülmüş görüntünün varyansı (sürekli).	Dalgacık Dönüştürülmüş görüntünün eğriliği (sürekli)	Dönüştürülmüş görüntünün basıklığı	Görüntünün entropisi (sürekli).	Sınıf (0-1)

Aşağıdaki Şekil 10'de dosyadan verinin okunmuş hali gösterilmektedir.



Şekil 10: data_banknote_authentication.csv.csv İsimli Dosyanın Okunması

Bu çalışmada Anaconda platformu üzerinde bilimsel çalışma ortamına olanak veren Spyder IDE'si yardımıyla Python yazılım dili kullanılarak yapılmıştır. Sürüm olarak Python 3.7 kullanılmıştır.

5. Sonuç

Geleneksel CART algoritması kullanılarak aynı veri seti ile sınıflandırma yapıldığında edilen n adet Accuracy, Max. Score, Mean Accuracy değerleri aşağıda Şekil 11'de gösterilmiştir.

```
In [5]: runfile('C:/Users/Batuhan/Desktop/temp.py', wdir='C:/Users/Batuhan/
Desktop')
Scores: [95.98540145985402, 96.35036496350365, 97.44525547445255,
95.98540145985402, 98.17518248175182]
Mean Accuracy: 96.788%
Max Scores: 98.175%
```

Şekil 11: Geleneksel CART Algoritması ile Sınıflandırma Sonuçlarının Gösterimi

Aynı veri seti kullanılarak yapılan algoritmik geliştirmeler sonucunda yeni CART algoritmasıyla sınıflandırma işlemi yapıldığında elde edilen sonuçlar Şekil 12'de gösterilmiştir.

```
In [4]: runfile('C:/Users/Batuhan/Desktop/pandas_calısması.py', wdir='C:/Users/
Batuhan/Desktop')
Scores: [96.71532846715328, 97.08029197080292, 96.71532846715328,
99.63503649635037, 96.71532846715328]
Mean Accuracy: 97.372%
Max Scores: 99.635%
```

Şekil 12: Yeni Geliştirilen CART Algoritması ile Sınıflandırma Sonuçlarının Gösterimi

Çapraz Validasyon aşamasında verilerin 5 alt sete ayrımı için yapılan algoritmik geliştirmeler sonucu daha doğru bir ağaç yapısı ile max score ve mean scores oranlarında performans artışı olduğu görülmektedir. Endüstriyel alanda sınıflandırma işlemi yapılmak istenildiğinde ön çalışma yapılarak, sınıflandırmaya etkisi en az ve en çok olan özelliklerin belirlenmesinin önemi ortaya çıkmıştır. Yeni algoritmik geliştirmeler sonrasında float veri tipindeki veriler kullanıldığında özellikle finans sektöründe müşteri risk analizi, fraud detection gibi alanlarda doğruluk oranlarının oldukça artacağı öngörülmüştür. Endüstriyel alanda elde edilecek başarının sınıflandırma algoritmalarına olan ilginin ve Ar-Ge çalışmalarının ilerleyen yıllarda daha da artacağı düşünülmektedir.

Finansal Destek

Yazar bu çalışma için herhangi bir finansal destek almamıştır.

Kaynakça

Akın, E. (2017). K-Fold cross validation. Retrieved from http://cagriemreakin.com/veri-bilimi/k-fold-cross-validation-1.html.

Albright, S. C., Winston, W. L. & Zappe, C. (2006). Data analysis & decision making *3.ed*. Australia: Thomson South-Western.

Arlot S & Celisse A. (2010). A survey of crossvalidation procedures for model selection. Statistics Surveys, 4, 40–79.

Berry, M. J. & Linoff, G. S. (2004). Data mining techniques: For marketing, sales, and customer relationship management 2. ed. USA:Wiley.

Bozan, F. (2010). CART (Classification and regression tree). Erişim adresi http://www.farukbozan.com/2010/01/cartclassification-and-regression-tree/

Deconinck, E., Hancock, T., Coomans, D., Massart, D.L.& Heyden, Y.V. (2005). Classification of drugs in absorption classes using the classification and regression trees (CART) methodology. Journal of Pharmaceutical and Biomedical Analysis, 39, 91–103.

Gordon, G. & Pressman, I. (1983). Quantitative decision-making for business 2. ed. USA: Prentice Hall International, Inc.

Hand D., Mannila H. & Smyth P. (2001). Principles of data mining. USA:MIT Press.

Kurt, I. , Ture, M. & Kurum, A. T. , (2008) Comparing performances of logistic regression, classification and regression tree, and neural networks for predicting coronary artery disease. Expert Systems with Applications, 34, 366–374.

Lewis R. J. (2000). An introduction to classification and regression tree (CART) analysis. Retrieved from https://www.researchgate.net/profile/Roger_Lewis6/publication/240719582_An_Introduction_to_Classification_and_Regression_Tree_CART_Analysis/links/0046352d3fb18f1740000000/An-Introduction-to-Classification-and-Regression-Tree-CART-Analysis.pdf.

Shaikh (2018). Cross Validation Explained: Evaluating estimator performance. Retrieved from https://towardsdatascience.com/cross-validation-explained-evaluating-estimator-performance-e51e5430ff85

Silahtaroğlu, G. (2008). Kavram ve algoritmalarıyla temel veri madenciliği. İstanbul:Papatya Yayıncılık Eğitim.

Teng, J. , Lin, K. & Ho, B. (2007). Application of classification tree and logistic regression for the management and health intervention plans in a community-based study. Journal of Evaluation in Clinical Practice, 13, 741–748.

Özgeçmiş

Batuhan Bilenler

8 Aralık 1992 tarihinde Malatya'da dünyaya geldi. 2011 yılında İstanbul Üniversitesi Bilgisayar Mühendisliği bölümüne girdi ve 2016 yılında mezun oldu. 2020 yılında İstanbul Aydın Üniversitesinde Bilgisayar Mühendisliği yüksek lisans programından mezun oldu. Halihazırda Anadolu Sigorta şirketinde Kıdemli Yapay Zekâ Mühendisi olarak çalışmaktadır.

Kutluk Özgüven

1990 yılında İhsan Doğramacı Bilkent Üniversitesi Bilgisayar Mühendisliği bölümünden mezun olmuştur. 1992 yılında Manchester Üniversitesi Bilgisayar Çevirisi (Yapay Zekâ) alanında yüksek lisans derecesi ve 2004 yılında İstanbul Üniversitesi Elektronik Medya Yönetimi alanında doktora unvanını almıştır. 2009 yılında doçent ve 2015 yılında ise profesör olmuştur.