Kırmızı Şarap Kalitesinin Makine Öğrenmesi Kullanılarak Tahmin Edilmesi

Predicting Red Wine Quality Using Machine Learning

Mehtap ÖKLÜ
Bilgisayar Mühendisliği
17110131052
Kahramanmaraş Sütçü İmam Üniversitesi
Kahramanmaraş, Türkiye
mehtap_oklu_06@hotmail.com

Banu KÖSE
Bilgisayar Mühendisliği
18110131011
Kahramanmaraş Sütçü İmam Üniversitesi
Kahramanmaraş, Türkiye
banukose1561@gmail.com

***Özetçe* —Şarap, çok eski tarihlerden beri tüketilen, batı toplumlarında mutfakla özdeşleşmiş olan alkollü bir içecektir. Farklı meyvelerle üretilebiliyor olsa da, şarap denince akla ilk gelen meyve üzümdür. Şarap, çok geniş bir skalaya sahiptir. Bu yüzden her şarabın kalite oranı farklıdır. Üzümle üretilen şaraplardan biri de kırmızı şaraptır. Kırmızı şaraba ait bazı özniteliklerin (alkol oranı, pH, klorür vb.) değerleri incelenerek şarabın kalitesi tahmin edilebilir. Bu değerlerin incelenmesi için de belirlediğimiz bazı yapay zeka algoritmalarını (KNN, SVC, Lojistik Regresyon, Naive Bayes, Karar Ağacı, Bagging ve Rastgele Orman Ağacı) kullandık. Kullandığımız algoritmalar genellikle sınıflandırma algoritmalarıdır. Bunun sebebi, şarabı kaliteli veya kalitesiz (0-1) olarak etiketlemek istememizdir.**

Anahtar Kelimeler—Kırmızı Şarap, Kalite, KNN, SVC, Lojistik Regresyon, Naive Bayes, Karar Ağacı, Bagging ve Rastgele Orman Ağacı

***Abstract*—Wine is an alcoholic beverage that has been consumed since time immemorial, identified with cuisine in western societies. Although it can be produced with different fruits, the first fruit that comes to mind when it comes to wine is grapes. Wine has a very Large scale. That is why the quality ratio of each wine is different. One of the wines produced with grapes is red wine. It should be noted that some attributes of red wine (alcohol content, pH, chloride, etc.) the quality of the wine can be estimated by examining its values. In order to examine these values, we used some artificial intelligence algorithms (KNN, SVC, Logistic Regression, Naive Bayes, Decision Tree, Bagging, and Random Forest Tree) that we also determined. The algorithms we use are usually classification algorithms. This is because we want to label the wine as good quality or poor quality (0-1).**

Keywords—Red Wine, Quality, KNN, SVC, Logistic Regression, Naive Bayes, Decision Tree, Bagging, and Random Forest Tree

I. GİRİŞ

Şarap, diğer adıyla mey. Parçalanmış veya parçalanmamış üzümün fermente edilmesiyle [1] üretilen, tescillenmiş veya tescillenmemiş, %9-15 oranda alkole sahip [2] bir içecektir. Tescillenmiş şaraplar, üretildiği coğrafi bölgenin bir işaretini taşır [3].

Şarap, bilinmeyen tarihlerden beri üretilen ve tüketilen bir içecektir. Elimizdeki bilgilere göre şu an şarabın bilinen doğum yeri Gürcistan'dır, tarihi ise MÖ 6000'lere kadar dayanmaktadır [2]. Şarap, birden fazla çeşide sahiptir. Bunlar; kırmızı, beyaz, rose, blush, likor şarabı vb. şeklindedir [1]. Bu makalede kırmızı şarabı inceleyeceğiz.

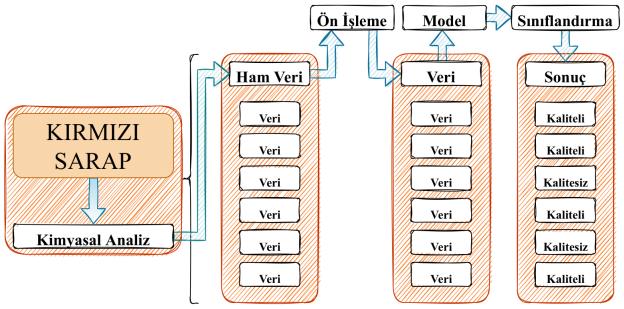
Şarap üretimi çok fazla emek isteyen, yorucu bir iştir. Gerek ilkel yöntemlerle, gerek de modern teknolojiyle şarap üretmek mümkündür. Şarap üretiminde yer alan bu teknolojinin ilerlemesiyle birlikte, şarap kalitesiyle alakalı kriterler de daha detaylı hale gelmiştir [1]. Bu kriterler ve parametreler(uçucu asitlik, artık şeker miktarı, alkol oranı, pH, klorür vb.) daha net bir şekilde incelenebildiği için şarabın kalitesiyle ilgili ölçümler de daha rahat yapılabilmektedir. Bu makalede de yapay zeka kullanarak kırmızı şarap kalitesini ölçmeye çalışacağız.

Kırmızı şarap üzerinde kimyasal ölçümler yapılarak bir takım veriler elde edildi. Bu veriler de Kaggle [4] üzreinde açık kaynak olarak paylaşıldı. Biz de bu veri setini edindik, üzerinde ön işleme yaptık ve üzerine makine öğrenmesi algoritmalarını kurduk. Bu makalemizde de makine öğrenmesi kullanarak kırmızı şarap kalitesini tahmin etmeye çalışacağız.

Bu projede "Red Wine Quality" isimli veri setin [4] Kaggle üzerinden temin ettik. Verinin %30'unu test için, %70'ini de eğitim için kullandık. Veri setinde 11 adet öznitelik, 1600 adet kayıt var. Bu öznitelikler:

- **fixed acidity (sabit asitlik):** Şarapla ilgili sabit, uçucu olmayan asitler. Kolayca buharlaşmazlar [5].
- **volatile acidity (uçucu asitlik):** Şarapta yüksek seviyelerde bulunduğunda sirke tadını ortaya çıkmasına neden olabilir [5].
- **citric acid (sitrik asit):** Şarapta az miktarda bulunduğunda daha taze ve fresh bir tat verir [5].
- **residual sugar (artık şeker):** Fermantasyon işleminden sonra arta kalan şeker miktarı, 1 g/L'den az olan şaraplar nadir bulunur. 45 g/L'den fazla şaraplar sweet(tatlı) olarak kabul edilir [5].
- **chlorides (klorür):** Şaraptaki tuz miktarıdır [5].
- **free sulfur dioxide (serbest kükürt dioksit):** Serbest form halinde bulunan kükürt dioksit, çözünmüş bir gaz ve bisülfit iyonu arasında dengede kalır.. Şarap oksidasyonunu önleme yanında büyümeyi (mikrobiyal) sağlar [5].
- **total sulfur dioxide (toplam kükürt dioksit):** Düşük seviyelerde, şarapta anlaşılması zordur fakat 50PPM'yi geçtiği zaman şarabın kokusunda ve tadında belirgin hale gelir [5].
- **density (yoğunluk):** Şarabın yoğunluğu 1.08 - 1.09 civarındadır. Bu durum da şarabın suya göre %8-9 daha yoğun olduğu anlamına gelir [6].
- **pH (asidik-bazik):** pH cetvelinde 0-14 arasında değerde şarabın asitlik ve baziklik değerini verir. Şaraplar pH cetvelinde 3-4 arasında yer alır [5].
- **sulphates (sülfür):** Antimikrobiyal ve antioksidan görevi gören kükürt dioksit gazının (SO2) seviyelerine katkıda bulunabilen bir katkı maddesidir [5].
- **alcohol:** Şarabın yüzde kaç alkol içerdiğini gösterir. Kırmızı şarapta alkol oranı %11-14 arasındadır [5].

Bu veri setinde kalite sütunu yani sonuç özniteliğinde 0-1 değerleri yer alıyor. Öznitelikler doğrultusunda şarabın yüksek veya düşük kaliteli olduğunu belirtiyor. Elimizdeki bu veri setinde en verimli şekilde etiketleme yapabilmek için; KNN, SVC, Lojistik Regresyon, Naive Bayes, Karar Ağacı, Bagging ve Rastgele Orman Ağacı Algoritmalarını kullanmaya karar verdik. Projemizi Python dilinde, Anaconda-Spyder editöründe yazdık.



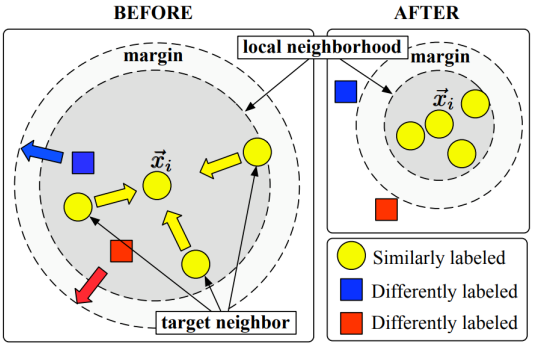
Şekil 1: Kırmızı Şarap Kalite Kontrol Süreci

Projemizin akış diyagramı Şekil 1'deki gibidir. Makalenin ikinci bölümünde, bahsettiğimiz algortimalar hakkında bilgiler sunulmaktadır. Daha sonraki bölümde ise sonuçlar yer almaktadır.

II. METODLAR

A. *K-Nearest Neighbor*

Gözetimli (denetimli) öğrenme metotlarından olup basit sınıflandırma işlemlerinde kullanılır [8] [9].Sınıflandırma çalışması yaparken elimizdeki veriler hakkında kısıtlı ön bilgiye sahip olduğumuzda tercih etmemiz gereken ilk algoritmalardan biridir [8].



Şekil 2: Örnek KNN Şeması [10]

Bu algoritmanın performansını etkileyen parametreler; uzaklık ölçütü, komşu sayısı(k) ve ağırlıklandırma yöntemidir [7]. Uzaklık ölçütü olarak Manhattan uzaklığı kullandık: **n** boyutlu düzlemdeki iki konum arasındaki farkların, mutlak değerlerinin toplamıdır. X-Y konumları arasındaki Manhattan uzaklığı: P=(x1, x2,. . . , xn) ve Q=(y1, y2,. . . , yn) olmak üzere, Eşitlik 1'e göre hesaplanır [7].

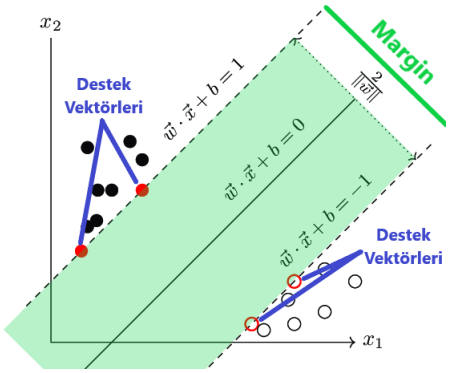
$$\sum_{i=1}^{n} |x_i - y_i| \qquad (1)$$

KNN algoritmasının aşamaları [9]:

- K değeri belirlenir
- Diğer konumlara olan uzaklık, Manhattan yöntemiyle hesaplanır
- Uzaklıklar sıralanarak en yakın komşular bulunur
- En yakın K adet komşunun kategorileri toplanır
- Toplam sonucunda ağırlıkta olan kategori seçilerek etiketleme yapılır

B. *Support Vector Machines (SVM)*

Türkçe adıyla Destek Vektör Makineleri (DVM), istatistiksel öğrenme teorisi geliştiricisi Vapnik tarafından geliştirilmiştir. Sınıflandırma, örüntü tanıma problemleri için kullanılır. Destek Vektör Makineleri, alanındaki birçok tekniğe göre daha yüksek başarı oranına sahiptir. Uygulama sırasında çekirdek fonksiyon seçimi ve parametre optimizasyonu çok önemlidir [11].



Şekil 3: Örnek SVM Şeması [12]

Şekil 3 üzerinde siyah ve beyaz olmak üzere iki sınıf mevcut. Gelecek verilerin sınıflandırılabilmesi için öncelikle sınıfları birbirinden ayıran bir doğru çizilir. Bu doğrunun -1 ve +1 değerleri arasında kalan yeşil kısım Margin bölgesidir. Margin ne kadar geniş olursa, sınıflar o kadar iyi ayırışıyor demektir. Bu işlemin formülü Eşitlik 2'deki gibidir.

$$\hat{y} = \begin{cases} 0, w^T \cdot x + b < 0, \\ 1, w^T \cdot x + b \geq 0 \end{cases} \quad (2)$$

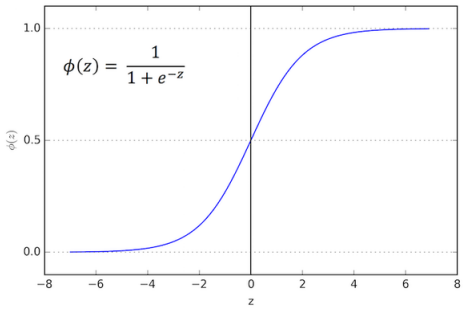
Elde ettiğimiz sonuç 0'dan küçük çıkarsa beyaz noktaların bulunduğu kısma yakın olacaktır. Sonuç 0'a eşit veya 0'dan büyük çıkarsa siyah noktaların bulunduğu kısma daha yakın olacaktır [12].

C. *Logistic Regression*

Lojistik Regresyon,bize sonucu 0 veya 1 şeklinde üreten, değişkenlerin modellenmesinde kullanılan algoritmadır. Örneğin kırmızı şarap kalitesini ele aldığımız zaman; elimizdeki veri setinde 0 kalitesiz, 1 kaliteli durumunu temsil eder. Yani Lojistik Regresyon algoritmasıyla predict edilen verilerin ait oldukları sınıfları tahmin edebiliriz [13].

$$f(x) = \frac{1}{1 + e^{-z}} \quad (3)$$

Lojistik regresyon, $(-\infty, +\infty)$ aralığındaki değerleri girdi olarak kabul eder [13].



Şekil 4: Örnek bir Lojistik Regresyon Şeması [13]

D. *Naive Bayes*

Navie Bayes algoritması sınıflandırıcı teorem olan Bayes'e dayanmaktadır [15]. Adını ingiliz matematikçi olan Thomas Bayes'ten almıştır [14]. Navie Bayes, elimizdeki örneklerin hangi sınıfa ait olduğunu, işlemler sonucu elde ettiği oranla tahmin eder. Navie Bayes'ın iki önemli kabulü vardır [15]. Bunlar:

- Elimizde bulunan niteliklerin hepsi aynı derecede önemlidir.
- Elimizde bulunan nitelikler birbirinden bağımsızdır.

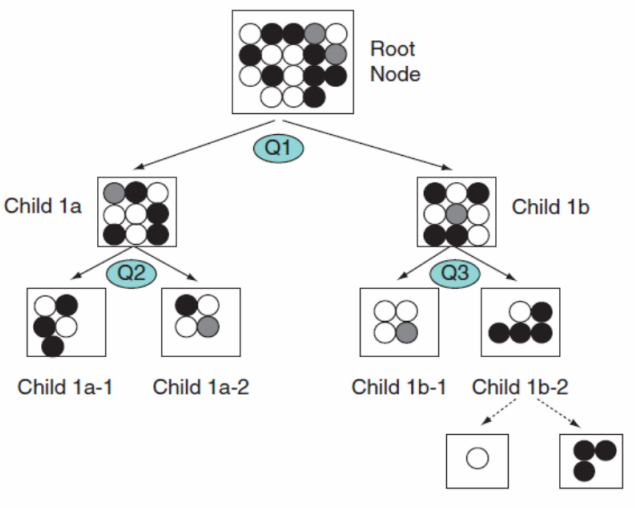
Koşullu olasılıklar arasında bağlantı kuran Bayes Teoremi, x özniteliğinin gerçekleşmiş olması durumunda C özniteliğinin gerçekleşmesi veya C özniteliğinin gerçekleşmiş olması durumunda x özniteliğinin gerçekleşmesi durumunu bize verir [15]. Bayes kuralının algoritması Eşitlik 4'deki gibidir. Eşitlikte yer alan değerler [14]:

- **p(x|Cj):** j sınıfından bir örneğin x olma olasılığı
- **P(Cj):** j sınıfının ilk olasılığı
- **p(x):** Herhangi bir örneğin x olma olasılığı
- **P(Cj|x):** x olan bir örneğin j sınıfından olma olasılığı (son olasılık)

$$P(C_j|x) = \frac{p(x|C_j)P(C_j)}{p(x)} = \frac{p(x|C_j)P(C_j)}{\sum_k p(x|C_k)P(C_k)} \quad (4)$$

E. *Decision Tree*

Karar ağaçları, en yaygın kullanılan gözetimli(denetimli) öğrenme algoritmalarından birisidir. Adından da anlaşılacağı üzere ağaç tabanlı öğrenmeye dayalıdır. Tüm problemlere uyarlanabilir [16]. Karar ağacı, kök düğüm değişkeninden başlayarak, ağaç dalı hiyerarşisi gibi dallanır [17].



Şekil 5: Örnek bir Karar Ağacı Şeması [17]

Entropi, verilerimizle alakalı belirsizliğin bir ölçüsüdür; dolayısıyla entropinin düşük olması tercih edilir. Verilerimizin tamamı sezgsel olarak aynı etikete sahipse, veri setinin entropisi düşüktür diyebiliriz [16].

$$H = -\sum p(x) \log p(x) \qquad (5)$$

Burada; p(x) belirli bir sınıfa ait grubun yüzde oranını, H ise entropiyi belirtmektedir. Karar ağacı, entropiyi en aza indirecek şekilde bölünme yapmalıdır. En iyi bölümlemeyi belirlemek için kullanılan bilgi kazancı, Eşitlik 6'daki gibi hesaplanır [16].

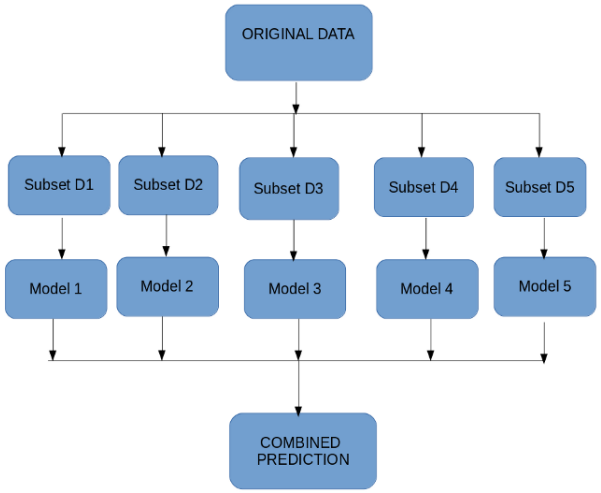
$$Gain(S, D) = H(S) - \sum_{V \in D} \frac{|V|}{|S|} H(V) \quad (6)$$

Burada; **S** orijinal veri kümesini, **D** ise kümenin bölünmüş bir parçasını temsil eder. Her **V** değeri, **S**'nin alt kümesidir [16].

F. *Bagging*

Türkçe adıyla torbalama yöntemi olarak adlandırılan Bagging algoritması 1996 yılında Breiman tarafından geliştirilmiştir. Bagging (torbalama) algoritmasının çalışma biçimi Şekil 6'da belirtilmiştir. Şekil 6'daki aşamaların açıklamaları [18]:

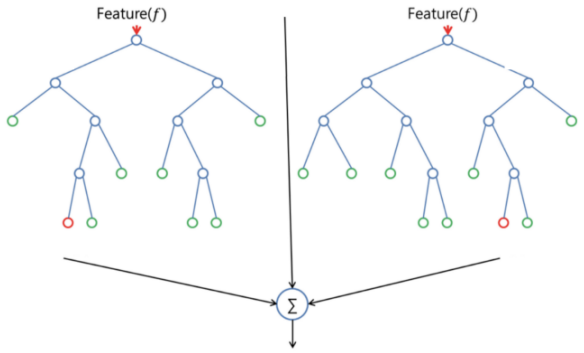
- **Bootstrap sampling:** Orijinal veri kümesi, alt kümelere bölünür.
- **Model training:** Bu alt kümelerin her birinde ayrı ayrı temel (zayıf) model oluşturulur.
- **Model forecasting:** Modeller birbirinden bağımsız ancak paralel olarak çalışır.
- **Result aggregating:** Tüm modellerin tahmin sonuçları birleştirilerek niai tahmin belirlenir.



Şekil 6: Örnek bir Torbalama Şeması [18]

G. *Random Forest Tree*

Türkçe adıyla Rastgele Orman Ağacı Algoritması, denetimli bir sınıflandırma algoritmasıdır. Adından da anlaşılacağı gibi, algoritmamız rastgele bir orman yaratıyor. Ormanda bulubab ağaç sayısı ile elde ettiğimiz çıktılar arasında doğrusal bir ilişki vardır. Ağaç sayısı arttıkça elde edilecek olan sonucun kesinliği de artar. Rastgele Orman Ağacı'nın Karar Ağacı'ndan farkı; kök bulma ve düğümleri bölme işlemlerinin çalışıyor olmasıdır [19]. Rastgele Orman Ağacı'nın şeması Şekil 7'de verilmiştir.



Şekil 7: Örnek bir Rastgele Orman Ağacı Şeması [20]

Rastgele Orman Ağacı, kullanıcıdan iki parametre alır: m parametresi en iyi bölünmeyi belirlemek için her düğümde kullanılan değişkenlerin sayısı, N geliştirilecek ağaç sayısı. Öncelikle eğitim verilerinin 2/3'ü kullanılarak önyükleme örnekleri oluşturulur. Geri kalan 1/3'lük kısım (OOB: out of bag) ise hataları test etmek için kullanılır [21].

Her bir düğümde **m** değişkenleri, tüm değişkenlerin arasından rastgele seçilerek en iyi dal belirlenir. **M** adet değişkenin karekköküne eşit olarak alınan **m** değişken, genellikle optimuma en yakın sonucu verir [21].

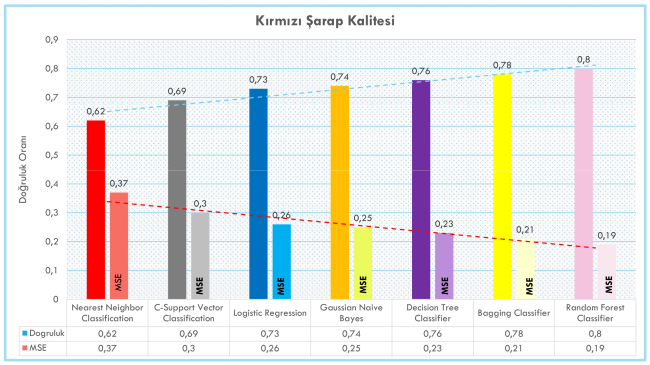
Sınıfların homojenliği $Gini$ indexi hesaplanarak ölçülür. $Gini$ indexi ne kadar düşükse, sınıf da o kadar homojendir. Bir düğümün alt $Gini$ indexi üst $Gini$ indexinden daha az olduğu durumlarda incelenen dal başarılı sayılır [21]. $Gini$ indexinin formülü Eşitlik 7'de verilmiştir. Formül değişkenlerinin temsil ettiği veriler de aşağıdaki gibidir;

- **T:** Tüm veri seti
- **pj:** Veri setindeki her bir verinin kendinden küçük ve kendünden büyük eleman sayılarına bölümü
- **n:** Seçilen verimiz

$$Gini(T) = 1 - \sum_{j=1}^{n} (p_j)^2 \qquad (7)$$

III. SONUÇLAR

Elimizdeki veri setinin $\%70$'ini, belirlediğimiz ve açıkladığımız algoritmaların eğitiminde kullandık. Bu eğitimler sonrasında, veri setimizinin geri kalan $\%30$'unu ise predict(tahmin) işlemi için kullanarak algoritma modellerimizi test ettik. Bu testler sonucunda modellerimizin, veri setimiz üzerindeki başarı ve hata oranlarını Şekil 8'deki grafiğe döktük.



Şekil 8: Tüm Algoritmaların Doğruluk & MSE Oranları

Kırmızı şarap kalitesinin tahmin edilebilmesi için, "Red Wine Quality" [4] isimli veri seti üzerinden Bagg, DT, KNN, LG, NB, RF ve SVM sınıflandırıcıları kullanılmıştır. Bu algoritmaların acc, MSE ve AUC değerleri, Tablo I'de verilmiştir.

	acc	**MSE**	**AUC**
Bagg	0.78	0.21	0.780
DT	0.76	0.23	0.761
KNN	0.62	0.37	0.629
LG	0.73	0.26	0.736
NB	0.74	0.25	0.743
RF	**0.80**	**0.19**	**0.799**
SVM	0.69	0.30	0.691

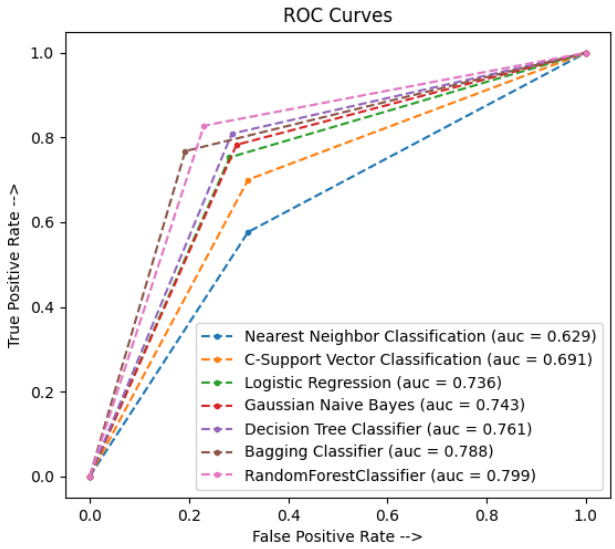
Tablo I: Bagging, DT, KNN, LG, NB, RF ve SVM sınıflandırıcılarının *acc, MSE ve AUC* ölçütlerinin ortalama değerleri

Tablo I incelendiği zaman, en yüksek doğruluk değerine (acc) sahip olan algoritmanın Random Forest (0.80) olduğunu görüyoruz. Ardından bunu takip eden algoritma ise 0.78 doğruluk oranı ile Bagging Classifier oluyor. Üçüncü sırada ise 0.76 doğruluk oranıyla Decision Tree algoritması yer alıyor. Geri kalan algoritmaların acc değerleri de Tablo I'de görülmektedir.

MSE (mean squared error/ortalama kare hatası) değerleri için Tablo I incelendiği zaman, en düşük MSE değerine sahip olan algoritmanın Random Forest (0.19) olduğunu görüyoruz. En yüksek MSE değerinin ise 0.37 oranıyla K-Nearest Neighboor algoritmasına ait olduğunu görüyoruz.

A. *Tüm Algoritmalara Ait ROC Eğrisi Grafiği*

Makine öğrenmesinde performans ölçümünde ROC eğrisinden de yararlanırız. Bu eğrilerin AUC(doğruluk) değerleri de mevcuttur ancak algoritmanın tahmin doğruluk skoruyla (acc) karıştırılmamalıdır [22]. Elimizdeki algoritmaların ROC eğrilerini Şekil 9'da verdik. Eğrilerin doğruluk oranları da Tablo I'de belirtilmiştir.

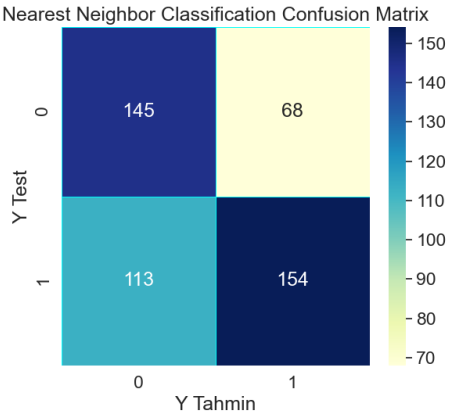


Şekil 9: Tüm Algoritmaların ROC Eğrileri ve AUC Değerleri

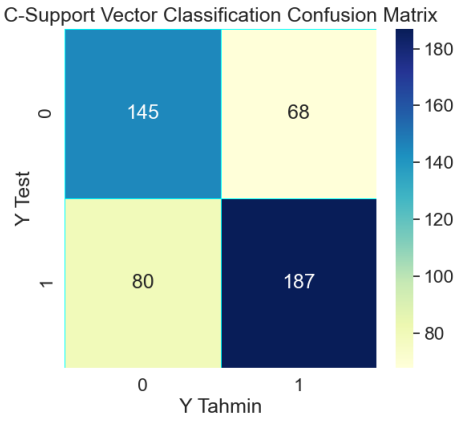
B. *Confusion Matrix (Hata Matrisi)*

Tablo I'de en yüksek doğruluk (acc) değerine (0.80) sahip olan Random Forest algoritması 0.19 değeri ile en düşük hata oranına sahiptir. Diğer algoritmalara baktığımız zaman elde edilen doğruluk (acc) ve hata (MSE) değerleri arasında ters orantı vardır. Yani acc değerimiz ne kadar yüksek olursa MSE değerimiz de o kadar düşük olmaktadır.

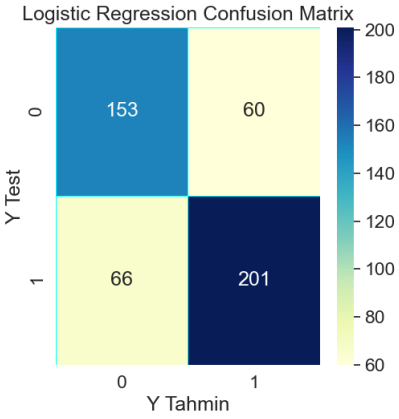
- $acc + MSE = 1$



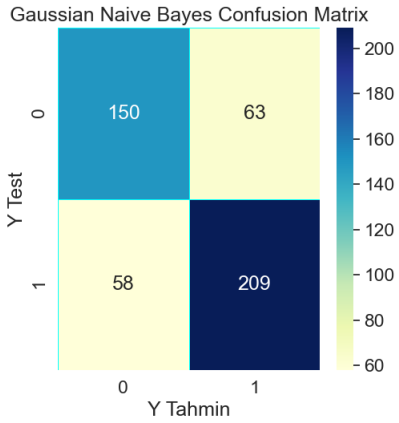
Şekil 10: K-Nearest Neighbor Hata Matrisi



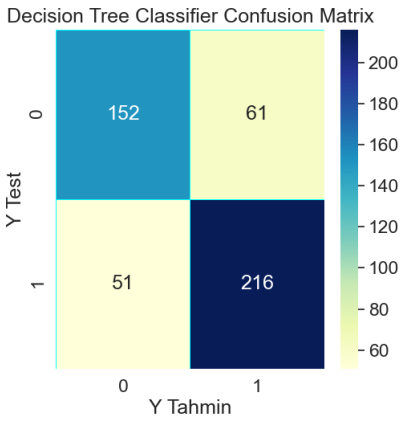
Şekil 11: Support Vector Machines Hata Matrisi



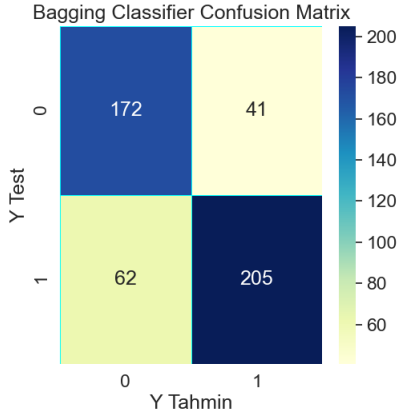
Şekil 12: Logistic Regression Hata Matrisi



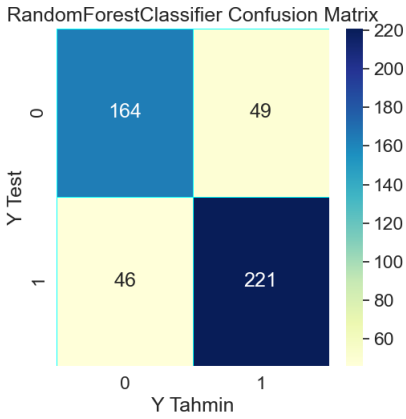
Şekil 13: Naive Bayes Hata Matrisi



Şekil 14: Decision Tree Hata Matrisi



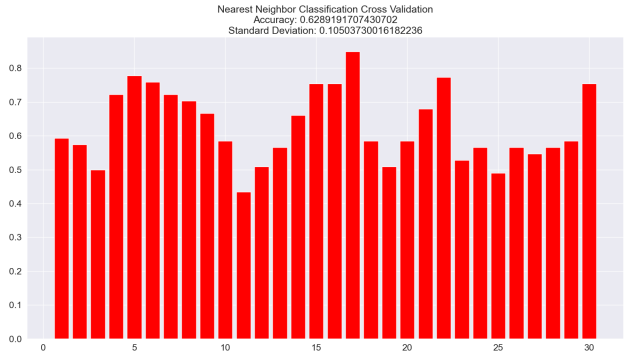
Şekil 15: Bagging Hata Matrisi



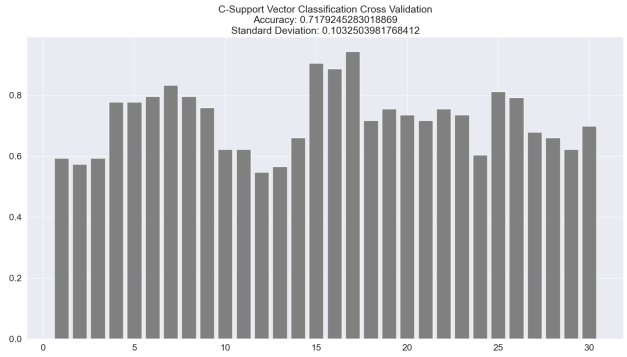
Şekil 16: Random Forest Tree Hata Matrisi

C. Cross-Validation (Çapraz Doğrulama)

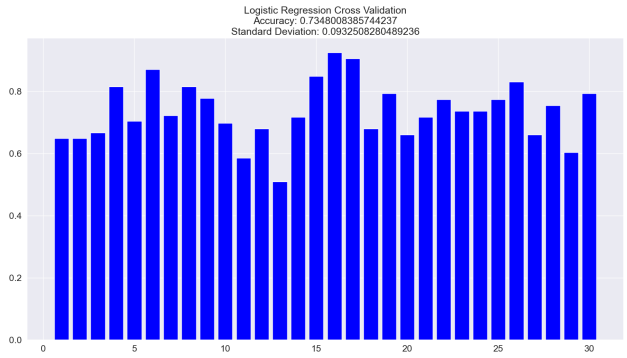
Sınıflandırma algoritmalarımızı bir kez eğittiğimiz zaman elde ettiğimiz verileri Tablo I'de belirttik. Elimizdeki Bagg, DT, KNN, LG, NB, RF ve SVM algoritmalarının üzerinde, **10 Kat Çapraz Doğrulama** yöntemini kullanarak 30 farklı koşma işlemi gerçekleştirdik. Bu koşma işlemlerinin tamamı rastgele ve birbirinden bağımsız olduğu için, farklı durumlarda algoritmaların nasıl çalıştığını da daha iyi anlamış olduk.



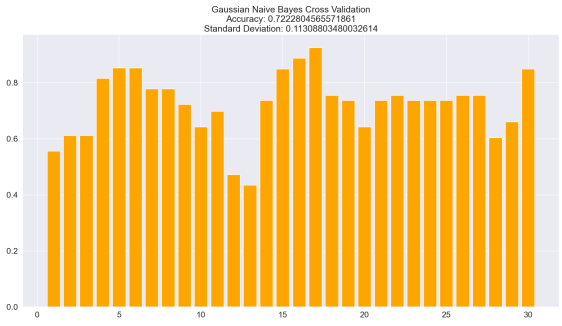
Şekil 17: K-Nearest Neighbor Cross-Validation



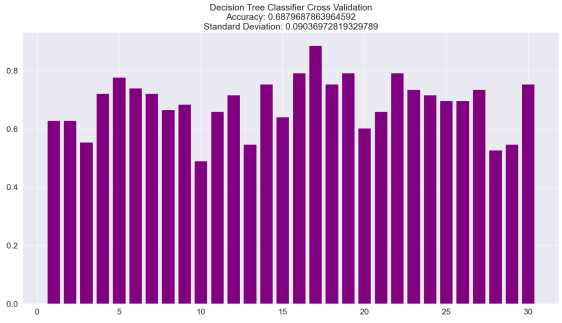
Şekil 18: Support Vector Machines Cross-Validation



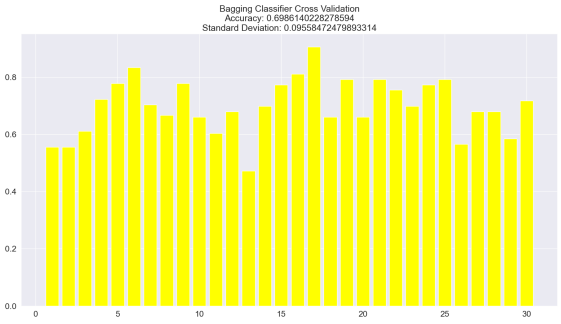
Şekil 19: Logistic Regression Cross-Validation



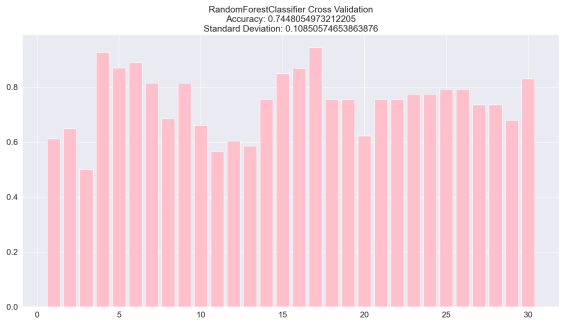
Şekil 20: Naive Bayes Cross-Validation



Şekil 21: Decision Tree Cross-Validation



Şekil 22: Bagging Cross-Validation



Şekil 23: Random Forest Tree Cross-Validation

Bagg, DT, KNN, LG, NB, RF ve SVM algoritmalarını **10 Kat Çapraz Doğrulama** yöntemi kullanarak eğittik, çıkan sonuçları ise Şekil 17, 18, 19, 20, 21, 22 ve 23'te görselleştirdik. Bu yöntem sonucunda;

- **Çalışma Sayısı**$(30)*$**Algoritma Sayısı**$(7) =$ **210**

adet sonuç elde ettik. Bu sonuçların ortalama değerlerini de Tablo II'de belirttik.

	acc	**St.Dev.**
Bagg	0.6986	0.0955
DT	0.6879	**0.0903**
KNN	0.6289	0.1050
LG	0.7348	0.0932
NB	0.7222	0.1130
RF	**0.7448**	0.1085
SVM	0.7179	0.1032

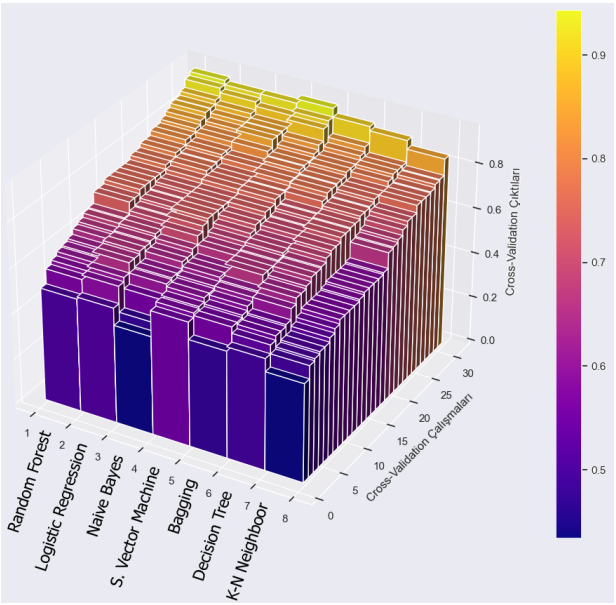
Tablo II: Bagging, DT, KNN, LG, NB, RF ve SVM sınıflandırıcılarının **Cross-Validation (10 Kat Çapraz Doğrulama)** sonuçlarının ortalama değerleri

Tablo II'deki değerlere baktığımız zaman en yüksek doğruluk ortalamasına sahip algoritmanın Random Forest (0.7448) ve en düşük standart devinasyon ortalamasına sahip algoritmanın da Decision Tree (0.0903) olduğunu görüyoruz.

Logistic Regression algoritmasının acc değerlerine baktığımız zaman diğer algoritmaların ortalamasının üstünde kaldığını ve $StandardDevination$ değerinin de diğer algoritmaların ortalamasının altında kaldığını görüyoruz. Bu senaryoda en verimli algoritmalardan birinin de Logistic Regression olduğunu söyleyebiliriz.

IV. SONUÇ

Bu projede kullanılan veri seti sayesinde kırmızı şarabın kalitesinin, ortalamanın üstünde veya altında olduğunu tahmin etmeye çalıştık. Bunu yapmak için, önceden belirlediğimiz algoritmaları kullandık ve sonuçlarını karşılaştırdık. Her algoritmada farklı doğruluk değerleri elde ettik. Bazı algoritmalar bu veri seti için daha yeterli olurken bazıları daha yetersiz kaldı. Algoritmaların veri setiyle verimli çalışabilmesi için, veri setinin detaylandırılması gerekir. Bunu da doğru özniteliklerin eklenmesi veya çıkartılması, örnek sayısının çoğaltılmasıyla sağlayabiliriz. Bu sayede kalite ölçümünü daha efektif hale getirebiliriz.



Şekil 24: Tüm Algoritmaların Cross-Validation Sonuçlarının 3D Grafiği

$X - ekseni$'ne baktığımız zaman algoritmalarımızın isimlerini, $Y - ekseni$'ne baktığımızda ise bu algoritmaların çalışma sayılarını görüyoruz. Elimizdeki $X - Y$ şeklindeki 2D ortama 3. bir boyut ekleyerek, bu algoritmaların çalışma sonuçlarını da $Z - ekseni$'ne eklemiş oluyoruz.

Elimizdeki bu 3D grafiği incelersek, en yüksek ortalamaya sahip algoritmanın Random Forest olduğunu anlayabiliriz. Çalışma sonuçlarını küçükten büyüğe sıralayarak bir rampa oluşmasını sağladık. Bu sayede rampanın yüksek kısmında kalan algoritmalar yüksek verimliliğer sahip algoritmalar oldu.

KAYNAKÇA

[1] DergiPark - Şarap Üretimi ve Kalite
dergipark.org.tr/sarap-uretimi-ve-kalite

[2] Wikipedia - Şarap
wikipedia.org/Şarap

[3] Resmi Gazete - Türk Gıda Kodeksi Şarap Tebliği
resmigazete.gov.tr/2009/02/20090204-12

[4] Kaggle - Red Wine Quality Dataset
kaggle.com/red-wine-quality

[5] Wine Quality Exploration
amazonaws.com/wine-quality-exploration

[6] Hydrometer Confusion
creativeconnoisseur.com/hydrometer-confusion

[7] Erdal Taşçı & Aytuğ Onan - KNN Algorithm
ab.org.tr/knn-algorithm

[8] Scholarpedia - K-Nearest Neighbor
scholarpedia.org/K-nearest_neighbor

[9] Erdinç Uzun - KNN Algoritması
erdincuzun.com/makine_ogrenmesi/k-nn-algoritmasi

[10] Large Margin Nearest Neighbor Classification
proceedings.neurips.cc/Margin_Nearest_Neighbor_Classification.pdf

[11] DergiPark - Destek Vektör Makinesi
dergipark.org.tr/Destek_Vektor_Makinesi

[12] Medium - Destek Vektör Makineleri
medium.com/deep-learning-turkiye/destek-vektör-makineleri

[13] Medium - Lojistik Regresyon
medium.com/lojistik-regresyon-makine-öğrenimi

[14] Başken Üniversitesi - Naive Bayes
baskent.edu.tr/Naive_Bayes.pdf

[15] Akademik Bilişim - Naive Bayes
ab.org.tr/Naive_Bayes.pdf

[16] Medium - Karar Ağaçları
medium.com/makine-öğrenimi-karar-ağaçlar\T1\i

[17] Sosyal Araştırmalar - Karar Ağacı
sosyalarastirmalar.com/decision-trees-algorithm

[18] Medium - Ensemble Learning
medium.com/ensemble-learning-bagging-ve-boosting

[19] Medium - Rastgele Orman Ağacı Algoritması
medium.com/rastgele-orman-algoritmas\T1\i

[20] DevHunter - Rastgele Orman Ağacı Algoritması
devhunteryz.com/rastgele-ormanrandom-forest-algoritmasi

[21] SlideShare - Rastgele Orman Ağacı Algoritması
slideshare.net/random-forest-algoritmas\T1\i

[22] Medium - ROC Eğrisi
medium.com/roc-eğrisi