İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ

WEB İÇERİK SINIFLANDIRMASI İÇİN MAKİNE ÖĞRENMESİ

YÜKSEK LİSANS TEZİ

Kenan Enes AYDIN

Bilişim Uygulamaları Anabilim Dalı

Bilgi ve Haberleşme Mühendisliği Programı

Mart 2020

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ

WEB İÇERİK SINIFLANDIRMASI İÇİN MAKİNE ÖĞRENMESİ

YÜKSEK LİSANS TEZİ

Kenan Enes AYDIN
(708171013)

Bilişim Uygulamaları Anabilim Dalı

Bilgi ve Haberleşme Mühendisliği Programı

Tez Danışmanı: Dr. Öğr. Üyesi Sefer BADAY

Mart 2020

ISTANBUL TECHNICAL UNIVERSITY★ INFORMATICS INSTITUTE

MACHINE LEARNING FOR WEB CONTENT CLASSIFICATION

M.Sc. THESIS

Kenan Enes AYDIN
(708171013)

Department of Applied Informatics

Information and Communication Engineering Programme

Thesis Advisor: Assis. Prof. Dr. Sefer BADAY

March 2020

İTÜ, Bilişim Enstitüsü'nün 708171013 numaralı Yüksek Lisans Öğrencisi Kenan Enes AYDIN, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı "WEB İÇERİK SINIFLANDIRMASI İÇİN MAKİNE ÖĞRENMESİ" başlıklı tezini aşağıda imzaları olan jüri önünde başarı ile sunmuştur.

Tez Danışmanı :	**Dr. Öğr. Üyesi Sefer BADAY** İstanbul Teknik Üniversitesi	.....
Jüri Üyeleri :	**Prof. Dr. M. Oğuzhan Külekçi** İstanbul Teknik Üniversitesi	.....
	Doç. Dr. Gökhan BİLGİN Yıldız Teknik Üniversitesi	.....

Teslim Tarihi : 28 Şubat 2020
Savunma Tarihi : 05 Mart 2020

ÖNSÖZ

Lisansüstü eğitimim süresince gerçekleştirdiğim çalışmalarımda ve derslerimde, tecrübesi, teorik bilgisi ve yol göstericiliği ile desteğini ve samimiyetini hiçbir zaman esirgemeyen saygıdeğer hocam Dr. Öğr. Üyesi Sefer BADAY'a, tez çalışmalarımda ve hayatımın her anında her zaman destekçim olan fedakâr eşim Ayşe AYDIN'a sonsuz teşekkürler.

Mart 2020

Kenan Enes Aydın
Elektrik-Elektronik Mühendisi

İÇİNDEKİLER

Sayfa

KISALTMALAR

BOW : Kelime Torbası (Bag of Words)

CSS : Basamaklı Stil Şablonu (Cascading Style Sheets)

CSV : Virgülle Ayrılmış Değerler (Comma Separated Values)

HTML : Hiper Metin İşaretleme Dili (HyperText Markup Language)

KNN : En Yakın k Komşu (K Nearest Neighborhood)

SVD : Tekil Değer Ayrışımı (Singular Value Decomposition)

SVM : Destek Vektör Makineleri (Support Vector Mahcines)

TF-IDF : Terim Frekansı-Ters Metin Frekansı ((Term Frequency-Inverse Document Frequency)

URL : Evrensel Kaynak Konumlandırıcıları (Universal Resource Locators)

ID3 : Yinelemeli Dikotomiser 3 (Iterative Dichotomiser 3)

CART : Sınıflandırma ve Regresyon Ağaçları (Classification and Regression Trees)

CHAID : Ki-kare Otomatik Etkileşim Detektörü (Chi-square Automatic Interaction Detector)

ÇİZELGE LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

WEB İÇERİK SINIFLANDIRMASI İÇİN MAKİNE ÖĞRENMESİ

ÖZET

Bu çalışmanın amacı, içerik türü bilinmeyen web sitelerinin sınıflandırılmasıdır. Metin içeriği baz alınarak uygulanacak analiz ve öğrenme teknikleriyle, mevcut ve her gün bir yenisi daha eklenen web siteleri sınıflandırılmakta ve kategori isimleriyle etiketlenmektedir. Çalışma çeşitli makine öğrenmesi algoritmaları kullanılarak web sitelerinden elde edilen metinlerin içerik türünü tahmin etmeyi amaçlamaktadır. Bu çalışmada kullanılan veri seti 30 bin web sayfasına ait HTML ham metinlerinin ön işlem aşamalarından geçirilmesi ile oluşturulan tekil anlamlı kelime gruplarını ihtiva eden metinlerden meydana gelmektedir. Bu metinler internet kullanım alışkanlıkları ve pedagojik yönelimler dikkate alınarak belirlenen 17 farklı kategori ismiyle etiketlenmiştir. Ardından bu metinler üzerinden n-gram ile öznitelik çıkarımı yapılmıştır. Çıkarılan bu öznitelikler üzerinden sistem Lojistik Regresyon, Destek Vektör Makineleri, Karar Ağaçları, Rassal Orman, Naif Bayes, Sinir Ağları, En Yakın Komşuluk ve Bire Karşı Tümü olmak üzere 8 farklı makine öğrenmesi tekniği üzerinde çapraz doğrulama ile test edilmiş ve sonuçlar farklı metrikler için incelenmiştir.

Web sitelerinin belirli kurallar altında sınıflandırılamaması bilgiye ulaşmayı zorlaştırdığı gibi farklı yaş gruplarından birçok kullanıcıyı internetin zararlı yüzü ile karşı karşıya bırakmaktadır. Arama motorlarının bazı eş sesli ve eşanlamlı sözcükler tekil olarak sorgulandığında gerçekte aranandan çok daha farklı sonuçlar önerdiği sıklıkla karşılaşılan bir durumdur. Bu olası yanlış yönlendirme durumu her yaştan internet kullanıcısı için tehlikeli sonuçlar doğurabilmektedir. Ülkemizde hizmet veren her bir İnternet servis sağlayıcısı bu duruma önlem almak amacıyla bir takım güvenli internet çözümleri sunmaktadır. Ancak bu çözümler kapsamlı olmadıkları gibi özelleştirilebilir de değildirler. Bunun yanında mahkemeler tarafından verilen engellenme kararlarının tüm zararlı siteleri kapsamaması ve yasaklanan bir siteye belli

bir süre sonra tekrar erişilebiliyor oluşu bir diğer eksikliği gözler önüne sürmektedir. Bu ve benzeri nedenler web sayfalarının yüksek hassasiyet ve başarım ile sınıflandırılmasını gerekli kılmaktadır.

Bu araştırma ile temel doğal dil işleme ve makine öğrenmesi teknikleri kullanılarak, Türkçe web sayfalarını da kapsayan, anlamlı ve anlamsız metinlerin dilden bağımsız olarak sınıflandırılacağı bir model önerilerek, ulaşılabilecek en yüksek sınıflandırma başarımının ortaya konması hedeflenmiştir. Dilden bağımsız sınıflandırma çalışmaları temelde önerilen 3 temel yöntemin uygulanmasını ve analizini içermektedir. Önerilen her bir yönteme ait strateji bu çalışmanın veri önişleme fazında adım adım takip edilmekte; sonuçlar öğrenme başarımı kriterlerine göre analiz edilerek en başarılı yöntem belirlenmektedir. Yöntemler çeviri ve doğal dil işleme işlemlerinin gerçekleştirilmesi ve bu işlemlerin gerçekleştirilme sıralarına göre farklılık göstermektedir. Etkisiz kelimelerin dile göre temizlenmesi ve diğer metin temizliği işlemleri önerilen tüm yöntemlerde ortak olarak uygulanmaktadır.

Bu çalışmada web sitelerini metin bazlı olarak sınıflandırılmasında karşılaşılan korelasyon ve aşırı öğrenme sorunlarını göz önünde bulundurarak, hata matrisleri değerlendirmesi ışığında sınıflar arası ilişkiler de incelenmiş ve sonuç olarak sunulmuştur. Sonuçlar web kaynaklı metin tabanlı sınıflandırmada yaklaşık %84 başarı oranı ile Lojistik Regresyon, Gradient Boosting ve Destek Vektör Makineleri tabanlı One VS Rest sınıflandırıcıların optimal parametreler ile ulaştıkları maksimum başarımı göstermektedir. Derin öğrenme ve doğal dil işleme teknikleriyle bu oranın iyileştirileceği öngörülmektedir.

MACHINE LEARNING FOR WEB CONTENT CLASSIFICATION

SUMMARY

This study aims to classify websites whose content type is uncertain. With the analysis and learning techniques based on the content of the text, the existing and added web sites added every day will be labeled with category names. The study aims to estimate the content type of texts from websites using various machine learning algorithms. The data set used in this study consists of texts containing singular meaningful word groups created by passing HTML raw texts of 30 thousand web pages through pretreatment stages. These texts are labeled with 17 different category names, which are determined taking into account the internet usage habits and pedagogical orientations.Then, feature extraction was performed with n-gram from these texts. With these extracted features, the system was tested with by cross validation on 8 different machine learning techniques including Logistic Regression, Support Vector Machines, Decision Trees, Random Forest, Naive Bayes, Neural Networks, Nearest Neighborhood, All Against One. The results were examined for 2 different metrics.

The fact that the websites cannot be classified under certain rules makes it difficult to access information and exposes many users of different age groups to the harmful side of the internet. It is a common occurrence that search engines propose results that are very different from what is actually searched when some synonyms and synonyms are questioned singularly. This possible misleading situation can have dangerous consequences for internet users of all ages. Every Internet service provider in our country offers a number of secure internet solutions in order to take precautions against this situation. However, these solutions are not comprehensive and are not customizable. In addition, the fact that the prohibition decisions made by the courts do not cover all harmful sites and that a banned site can be accessed again after a certain period of time is another deficiency. These and similar reasons require the web pages to be classified with high precision and performance.

In this study, the effects of word-based text classification and classification algorithms on unrelated texts were observed. The processes cover all the training, testing, cross-validation phases of a machine learning process from the provision of a web page raw data.

The raw text data of a website is saved as a file containing the website name. After all the addresses in the classified website list are queried and saved, all these text files are combined under a single file. This file is a plain text file containing a list of data.

This master file now represents the concept of data set, the first major stage of the machine learning process. In order to be valid at all times in the studies studied with the data set, the data are divided into two parts as education and test data with a common rate. After this process, 4 different finite-sized series named X_train, X_test, y_train and y_test are obtained.

The X_train object is a series of words, the source of which is the raw text document of the websites, where many operations like text cleaning and normalization will be applied. The y_train object, on the other hand, is labels that are considered classes of all these text series. These labels form the horizontal column of the high-dimensional matrix that will occur when texts are word-based vectorized. In simpler terms, X_train and y_train generate the input of the algorithm during the learning process. X_test, on the other hand, is the object that tests the model's predictive ability after learning is completed and has a content similar to X_train. The critical point here is that the test and training data should not overlap. This will cause the prediction process to produce misleading results and the algorithm to memorize rather than learn.

With this research, it is aimed to reveal the highest classification performance that can be achieved by using a basic natural language processing and machine learning techniques, by suggesting a model that includes Turkish web pages and where meaningful and meaningless texts are classified independently from the language. Language independent classification studies basically involve the application and analysis of the 3 main methods proposed. The strategy of each proposed method is followed step by step in the data preprocessing phase of this study; The most successful method is determined by analyzing the results according to the learning performance criteria. The methods differ according to the realization of the translation and natural language processing processes and the order of these processes. The cleaning of

ineffective words according to language and other text cleaning processes are applied in all recommended methods.

In this study, considering the correlation and excessive learning problems encountered in the classification of websites as text-based, the relationships between the classes are also examined and presented as a result. The results show a maximum success rate of one VS Rest classifiers based on Lojsitic Regression, Naive Bayes and Support Vector Machines based on optimal parameters with a 83% success rate in web based text classification. Deep learning and natural language processing techniques are expected to improve this ratio.

1. GİRİŞ

İnternetin büyümesi ve yaygınlaşarak her geçen gün yeni amaçlar için kullanılması farklı içerikteki web sitelerinin sayılarının hızla artmasına sebep olmaktadır. Faydalı içeriğe sahip olanların yanı sıra zararlı içeriğe sahip birçok web sayfası bulunmaktadır. Bu web sayfalarına her gün bir yenisi daha eklendiğinden içeriklerin türleri insanlar tarafından elle tespit edilememekte ve idari tedbir uygulamaları bu sitelerin zararlarından korunma noktasında yetersiz kalmaktadır.

Çalışmanın amacı makine öğrenmesi yöntemleri kullanılarak web sitelerinin metin bazlı olarak sınıflandırılmasıdır. Çalışmalar sonucunda geliştirilen algoritmalar ile internet dünyasına her gün bir yenisi daha eklenen ve içeriği belirsiz olan web sayfalarının sınıflandırılması hedeflenmektedir. Bu sayede zararlı web sayfalarının engellenmesi ve kategori yapısı sayesinde internet ortamında aranan sonuca daha kolay erişilebilmesi de sağlanmış olacaktır.

Arama motorları, web sitelerini metin bazlı olarak sınıflandırmakta ve bunu gerçekleştirirken web sitelerine ait meta verileri kullanmaktadırlar. Ancak bu meta veriler dilden dile farklılık göstermektedir. İşbu araştırma konusunun literatürde dolduracağı en önemli boşluk ise şu ana kadar Türkçe sitelerini de kapsayan herhangi bir sınıflandırma çalışması yapılmamış olmasıdır. Makine öğrenmesi alanında geniş çaplı çalışmalar gerçekleştiren çok uluslu global şirketlerin çalışmaları da dahil olmak üzere, yapılan içerik bazlı sınıflandırma çalışmaları bölgesel web sayfaları için yetersiz kalmakta hatta yanıltıcı derecede hatalı sonuçlar vermektedir.

Bu araştırma ile doğal dil işleme ve makine öğrenmesi teknikleri bir arada kullanılarak, Türkçe web sayfalarını da kapsayan, anlamlı ve anlamsız metinlerin dilden bağımsız olarak sınıflandırılmasında ulaşılabilecek en yüksek başarımın ortaya konması hedeflenmektedir. Literatürde çeşitli metin ve web içerik sınıflandırma problemleri için birçok çalışma yapılmıştır. Bunlara örnek olarak web sayfalarını anlamsal bölümlere ayrılması [2], konu sınıflandırma [4], grammer bazlı sınıflandırma [3], ikili ve çoklu içerik sınıflandırma [1] verilebilir.

Bu çalışmalarda her problem türü için çeşitli kelime vektörleştirme ve sınıflandırma yöntemleri önerilmiştir. En çok kullanılan sınıflandırma yöntemleri; Naive Bayes sınıflandırıcı algoritması [1], Rassal Orman sınıflandırıcı algoritması [1], Destek Vektör Makineleri sınıflandırıcı algoritması [1, 6], Lojistik Regresyon sınıflandırıcı algoritması [1], Karar Ağaçları sınıflandırıcı algoritması [5] ve Naive Bayes sınıflandırıcı algoritmasıdır [1,4,5].

Bununla birlikte literatürde farklı web sayfası ham metin türleri için, metin bazlı sınıflandırma ve bu sınıflandırma yöntemlerinin detaylı bir karşılaştırması bulunmamaktadır. Ayrıca literatürde web sayfalarının metin bazlı sınıflandırılması problemi için, web sayfalarından anlamlı metin temini ve bu metinler ile veri seti oluşturulması üzerine detaylı bir çalışma bulunmamaktadır. Literatürdeki çalışmaların önemli bir eksiği de sınıflandırma, boyut azaltma ve vektörizasyon uygulamalarının kurallı ve düzenli metinler üzerinde uygulanmasıdır. Bu çalışmada, bu boşluğu doldurmak amacıyla literatürdeki birçok yöntem ve bu çalışmanın elde etmiş olduğu yöntemler, yine bu çalışma sonucu elde edilen ve çalışmanın önemli bir parçasını oluşturan, web sayfalarının ham metinlerinden oluşturulan veri seti üzerinde uygulanarak karşılaştırılmıştır.

1.1 Literatur Araştırması

"Metin Madenciliği ve Makine Öğrenmesi İle İnternet Sayfalarının Sınıflandırılması" konulu çalışmada hedef İnternet sayfalarını sınıflandırmaktır. Çalışmanın sonunda girdi olarak bir alan adı verildiğinde, oluşturulan model sayesinde bu alan adına ilişkin bir sınıf bilgisinin geri dönüş olarak alınması amaçlanmıştır. Bu çalışmada, farklı makine öğrenmesi yöntemleri ve yapay sinir ağları kullanılarak İnternet sitesi sınıflandırma problemi incelenmiştir. Bu sınıflandırma probleminin çözümü için, İkili Sınıflandırma ve Çok Sınıflı Sınıflandırma olarak iki farklı yaklaşım uygulanmış, her iki yaklaşım da çalışma kapsamında toplanan İnternet siteleri üzerinde test edilip, performansları karşılaştırılmıştır [1]. Çalışmamız bu çalışmadan farklı olarak yalnızca İngilizce web sayfalarını değil her dildeki web saflarını veri setinde ihtiva etmektedir. Doğal dil işleme algoritmalarıyla sitelerin dili tespit edilip veri ön işleme aşamaları bu doğrultuda gerçekleştirilmiştir. Çalışmamızın bu çalışmadan farklı bir diğer yönü ise ham web metinlerden elde edilmiş anlamlı metinlerdeki tüm kelimelerin kullanılmamasıdır. Anlamsız ve kural dışı, kelime ve ses grupları nihai metin

dosyasına dahil edilmemiştir. Ayrıca çalışmada veri seti oluşturulurken kullanılan sınıflandırılmış web sayfaları listesi bir değil birden fazla kaynaktan alınıp karşılaştırılarak kullanılmış ve güvenilirliği arttırılmıştır.

"Classification of Web Elements Using Machine Learning" konulu çalışmada web sayfalarını anlamsal bölümlere ayırmanın yeni bir yolu sunulmuştur. Bu, belirli bir web sitesini ayrıştıran, sitenin web öğelerinin tüm ilgili özelliklerini toplayan ve her web öğesinin görüntülerini yakalayan bir prototip oluşturarak gerçekleştirilir. Toplanan veriler, web sitesi bölümlerini sınıflandırmak için bir makine öğrenme modelini eğitmek için kullanılan bir eğitim ve test verisi oluşturmak için kullanılmıştır. Üç farklı makine öğrenme algoritması, rastgele ormanlar, gradyan artırma makineleri ve sinir ağları incelenmiş ve test edilmiştir [2].

"Türk dilinin gramer özelliklerini kullanarak web tabanlı haber makalelerinin metin sınıflandırması" konulu çalışmada sınıflandırma başarı oranlarından ödün vermeden Türkçe'nin gramer kuralları kullanılarak öznitelik vektörünün boyutunun nasıl düşürüleceği açıklanmıştır. Bu seçimin boyut azaltma yöntemlerinin de yardımıyla özellik vektörünü %97'ye kadar düşürdüğü belirtilmiştir. Azalan özellik vektörünü kullanarak, daha iyi başarı oranları genellikle Naive Bayes, SVM, C 4.5 ve RF sınıflandırma yöntemlerinden elde edileceği ve elde edilen en iyi performansın, Naive Bayes yöntemi kullanılarak elde edilen %92,73'lük performans olacağı belirtilmiştir [3].

"Web içerik madenciliği ve konu sınıflandırılması" konulu çalışmada web ortamları için, Google arama motoru ile bütünleşik, bir konu sınıflandırma sistemi geliştirilmiştir. Ayrıca metin sınıflandırma da kullanılan Navie Bayes, Destek vektör makineleri, KNN yakın komşuluk algoritması ve karar ağacı algoritmalarının sınıflandırma performansı test edilmiş ve sonuçlar karşılaştırılmıştır. Yapılan analiz sonucunda sayfanın gerçekte hangi konu ile ilgili olduğu tahmin edilmiştir. Yapılan bu tahminlerin, web ortamında, kullanıcıların aradığı bilgilere daha kestirme ulaşmasına yardımcı olacağı düşünülmektedir [4].

"Yapay Sinir Ağları ile Web İçeriklerini Sınıflandırma" konulu çalışmada web sayfalarının belirlenen konulara göre sınıflandırılabilmesi için, Çok Katmanlı (MLP) yapay sinir ağı modeli kullanılmıştır. Özellik vektörü içeriğinin seçimi, yapay sinir ağının eğitilmesi ve son olarak web sayfalarının doğru kategorize edilmesi için bir

yazılım geliştirilmiştir. Bu yaklaşımın, elektronik ortamlarda bilgilerin kolaylıkla ve yüksek doğrulukla sınıflandırılması, web ortamlarında doğru içeriğe ulaşılması ve birçok güvenlik açığının giderilmesine katkılar sağlayacağı değerlendirilmektedir [5].

"Fast web page classification without accessing the web page using machine learning techniques" konulu çalışmada web sayfasına erişmeden bile web sayfalarını tek başına URL özelliklerine göre sınıflandırmanın mümkün olduğunu ve web sayfalarını URL özelliklerine göre sınıflandırmak için Destek Vektör Makinesi sınıflandırma algoritmasının en iyi sınıflandırıcı olduğu belirtilmektedir [6].

"Automated Classification of Web Sites Using Naive Bayesian" konulu çalışmada, "Bir web sitesinin ana sayfasının ifade gücü, kuruluşun yapısını tanımlamak için kullanılabilir." varsayımı ile Naïve Bayesian makine öğrenme algoritması kullanılarak web siteleri ana sayfalarının içeriğine göre sınıflandırılmaya çalışılmaktadır [7].

1.2 Kullanılan Python Kütüphanaleri

Python programlama dili (R. v. Guido,1991) makine öğrenmesi uygulamaları için kapsamlı paketleri barındırması ve bu paketlerin kolay kullanılmasından dolayı bu alanda oldukça popülerdir.

1.2.1 Doğal dil araç seti (NLTK)

Doğal dil araç seti (Natural Language Toolkit) Python programlama dilinde yazılmış bir doğal dil işleme kütüphanesi ve program kitidir.

1.2.2 Scikit-Learn

Scikit-Learn Python programlama dili için yazılmış bir makine öğrenmesi kütüphanesidir. Doğrusal regresyon, lojistik regresyon, karar ağaçları, rassal orman (random forest) ve birçok temel yapay öğrenme yöntemini içerisinde barındırmaktadır.

1.2.3 Chardet

Chardet, Python programlama dilinde yazılmış, işlevi bir metin üzerinde metnin dilini, karakter kodlamasını ve bu tespitlerin güven yüzdesini hesaplamak olan bir karakter kodlama detektörü olarak çalışır.

1.2.4 Urllib

Bu modül, World Wide Web'de veri toplamak için, urlopen () işlevini kullanır, dosya adları yerine Evrensel Kaynak Konumlandırıcıları (URL'ler) kabul eder. Alınan URL internette sorgulanır ve fonksiyon ham HTML verisini döndürür.

1.2.5 Numpy

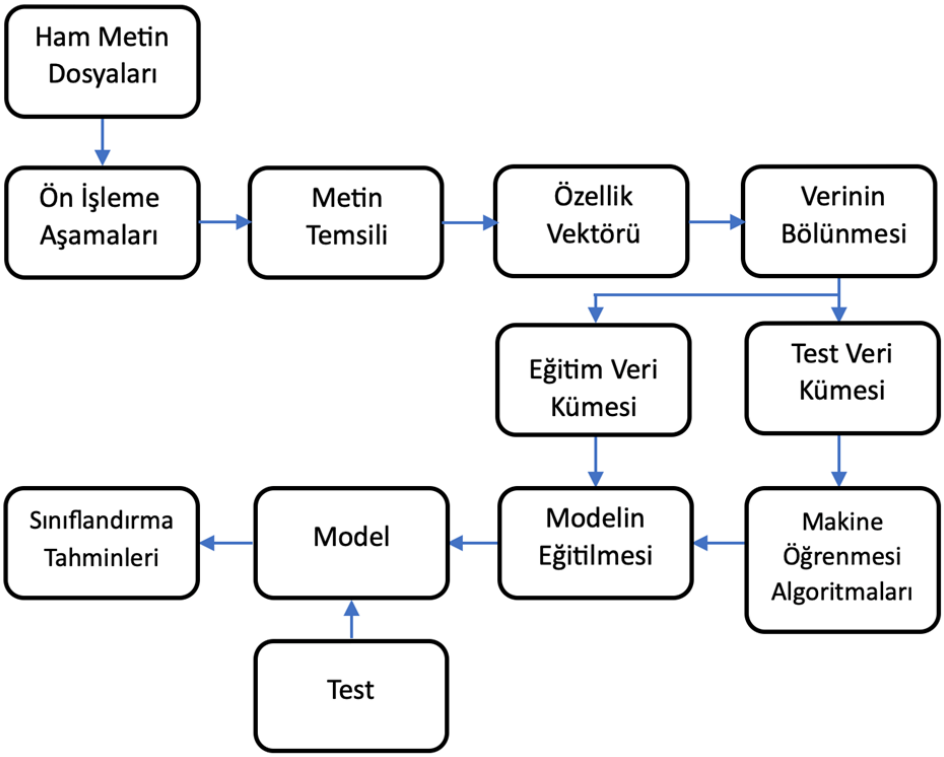
NumPy bilimsel hesaplamalarda kullanılan temel bir pakettir. C++ programlama dilinde yazıldığı için hesaplamaların hızlı şekilde yapılmasını sağlayan bir matematik kütüphanesidir.

1.2.6 Pandas

Veri işlemede ve makine öğrenmesi çalışmalarında sıklıkla kullanılan Pandas, seriler ve veri çerçeveleri işlevleri ile oldukça efektif bir kullanıma imkân veren bir Python modülüdür.

2. ÖNERİLEN MODEL

Metin bazlı sınıflandırma çalışmalarında metinler, elemanlarının her biri bir kelimeden oluşan yüksek boyutlardaki dizileri veri olarak kabul eden ve bu nedenle çok sayıda özniteliği sahip olan bir çalışma alanıdır. Sınıflandırılmak istenen verinin öznitelikleri kelimelerdir ve büyük belgelerde kelimelerin sayısı çok yüksek sayılara ulaşmaktadır. Daha küçük boyutlu, birkaç yüz kelimeden oluşan metinlerde hedeflenen sınıflandırma çok daha başarılı şekilde yapılıyor olsa da roman, makale gibi uzun metinler ve büyük internet verileri üzerinde bu işlev son derece karmaşık bir işlem haline dönüşmektedir.



Şekil 2.1 : Sınıflandırma süreci ve modelin inşası

Sınıflandırma ve regresyon, gelecekteki veri eğilimlerini tahmin eden modeller kurabilen önemli analiz yöntemleridir. Sınıflandırma kategorik sonuçlar veren değerleri tahmin ederken, regresyon süreklilik gösteren verilerin tahmin edilmesinde kullanılır. Sınıflandırmada bir veri öğesi önceden belirlenmiş olan sınıflardan birisine

yerleştirilir. Ancak regresyonda veri öğesi gerçek değerli bir tahmini değişkene eşlenir. [9]

Bu çalışmada kuralsız metinler üzerinde kelime bazlı metin sınıflandırma ve sınıflandırma algoritmalarının metin türü üzerindeki etkileri gözlemlenmiştir. Süreçler bir web sayfası ham verisinin temininden itibaren bir makine öğrenmesi sürecinin eğitim, test, çapraz doğrulama aşamalarının tümünü kapsamaktadır.

Bir web sitesine ait ham metin verisi hafızaya web sitesi ismini ihtiva edece bir dosya olarak kaydedilir. Sınıflandırılmış web sitesi listesindeki tüm adresler Python programlama dilinin urllib kütüphanesinin işlevleri ile sorgulanıp kaydedildikten sonra tüm bu metin dosyaları tek bir CSV dosyası altında birleştirilir. Comma Separated Values (CSV) – Virgülle Ayrılmış Değerler dosyası, bir veri listesi içeren düz metin dosyasıdır. Bu dosyalar genellikle farklı uygulamalar arasında veri alışverişinde kullanılır. Makine öğrenmesi çalışmalarında veri setleri çoğunlukla CSV dosyalarıyla kullanılırlar.

Bu ana dosya artık makine öğrenmesi sürecinin ilk ana aşaması olan veri seti kavramını temsil eder. Çoğunlukla bu dosyaların boyutları bellidir. Ancak birden fazla veri seti ile çalışılan araştırmalarda her zaman geçerli olması için ortak bir oranla veriler eğitim ve test verisi olmak üzere ikiye ayrılırlar. Burada Python'ın sklearn kütüphanesi altındaki eğitim ve test verisi bölmeye yarayan işlev istenilen oranın parametre olarak girilmesi ile kullanılır. Bu işlem sonrasında genel de X_train, X_test, y_train ve y_test adında 4 farklı sonlu boyutlu seri elde edilir. X_train metin temizliği, normalleştirme vb. birçok işlemin uygulanacağı, kaynağı web sitelerinin ham metin belgesi olan ve kelimelerden oluşan bir seridir. y_train tüm bu metin serilerinin sınıfları kabul edilen etiketlerdir ve metin kelime bazlı vektörleştirildiğinde oluşacak yüksek boyutlu matrisin yatay sütununu oluştururlar. Daha basit bir ifade ile öğrenme işlemi sırasında X_train ve y_train algoritmanın girdilerini oluşturur. X_test ise öğrenme tamamlandıktan sonra modelin tahmin yeteneğini sınayan ve X_train'e benzer içerikte olan nesnedir. Burada kritik nokta test ve eğitim verilerinin ortak veri içermemesi gerektiğidir. Bu tahmin işleminin yanıltıcı sonuçlar vermesini ve algoritmanın öğrenmek yerine ezber yapmasına sebep olacaktır.

Vektörleştirme adımından sonra özellik vektörü çıkartılmış olan eğitim ve test verisi makine öğrenme algoritmalarına girdi oluşturmaya hazır hale getirilmiştir. Bu

aşamada sınıflandırma algoritması seçilir. Algoritmaların çalışma prensipleri ve algoritmik temelleri gereği eğitim aşamasının başında farklı parametreler alırlar. Bu parametreler sahip olduğumuz veri setinin özelliklerine göre değişen ve sistemi en iyi şekilde eğitip en başarılı sınıflandırmayı yapmamıza olanak sağlayan kurallardır. Kurallarını belirlenen algoritmanın sıradaki işlemi eğitim verisini girdi alarak öğrenmeye başlamasıdır. Eğitim tamamlandıktan sonra X_test modelin tahmin fonksiyonuna girdi teşkil eder. Bu fonksiyonun tahmin çıktısı tahminler adını verebileceğimiz bir nesneye kaydedilip sınıflandırma başarısını ölçmek için y_test nesnesi ile algoritmanın değerlendirme fonksiyonuna girdi oluşturur. Bu fonksiyonun çıktısı sınıflandırma işleminde hangi sınıf için ne kadar başarı elde ettiğimizi gösterecek olan nihai sonuçtur. Bu noktadan sonra başarımın artırılması için çapraz doğrulama ile optimum parametreler belirlenir. Çapraz doğrulama işlemi sınıflandırma algoritmasına tipine bağlı olarak, her bir parametre için uygun aralıklarda seçilen birden fazla değişkendir. Algoritma sırasıyla bu parametreleri kullanrak yeniden sistemi eğitir ve tüm sonuçları karşılaştırıp en yüksek başarımı sağlayan parametre setini bize sunar. Bu mevcut algoritma, veri seti ile ulaşılabilecek en iyi başarımlardan birini gösterir.

2.1 Metin Sınıflandırma

Metin temsili adımı, belgedeki tüm metin içeriğinin, bir sınıflandırıcı tarafından anlaşılacak ve sınıflandırılabilecek türde bir vektöre dönüştürülmesi işlemidir. Metin temelli bir belge de her bir terim farklı bir öneme sahiptir. Terim ağırlıklandırma yöntemleri de bu terimlere, metin sınıflandırma performansını artırmak için farklı kriterleri göz önünde bulunduracak şekilde uygun ağırlıklar verirler. [8]

2.1.1 Metin sınıflandırmasında terim ağırlıklandırma

Metin kategorizasyonundaki ana konulardan biri belgelerin temsilidir. Son yıllarda metin sınıflandırması, belgelerin indekslenmesi, duyarlılık analizi, veri madenciliği, kalıtsal hastalıkların tespiti ve benzeri birçok alanda kullanılmaktadır. Bu sınıflandırma alanı içerisinde barındırdığı bir takım değerlendirme ve yorumlama kriterleri ile farklı amaçlar için kullanılır hale gelmiştir. Metin sınıflandırmasında ağırlıkların değerlendirme kriterleri üç başlık altında toplanacak olursa:

Terim sıklığı faktörü (Term Frequency Factor): Bir belgede bir terimin birden fazla oluşu belgenin içeriğiyle ilgili bilgi vermesi;

Koleksiyon sıklığı faktörü (Collection frequency factor): Bir belgede bir terimin sık bulunmamasının belge içeriğini daha iyi ayırt etmesi;

Uzun belgeler, kısa belgelerden daha önemli değildir, normalizasyon (Edgar F. Codd, 1970), belgelerin uzunluğunu eşitlemek için kullanılır. [10]

2.1.2 Denetimsiz terim ağırlıklandırma yöntemleri

Denetimsiz terim ağırlıklandırma şemaları genelde belgelerin kategori etiketlerini dikkate almadan bilgi çıkarımı üzerinden oluşturulur. Terim ağırlıklandırmadaki üç kriteri kusursuz bir şekilde içeren en popüler denetimsiz terim ağırlıklandırma yöntemi tf-idf'tir (Salton ve Buckley 1988). Bilgi erişiminde yaygın olarak kullanılmıştır ve sonuç olarak metin kategorizasyonunda çalışmalarında yaygın olarak tercih edilmektedir.

2.1.3 Denetimli terim ağırlıklandırma yöntemleri

Genellikle metin sınıflandırması, eğitim verisi olarak önceden kategorileri tanımlanmış belgelerdeki bilgileri kullandığı için denetimli bir öğrenme şeklidir. Özellik çıkarımı ve performans artırımı için eğitim verisindeki bu etiketli veriler oldukça önemlidir ve bu veriler sınıflandırıcının modellenmesinde de kullanılmaktadır. Özellik seçiminin temel amacı, sınıflandırma görevi için en uygun ve ayırt edici özellikleri seçmektir. Bu seçim bir puanlama sistemiyle önem sırasına dönüştürülür. Yüksek puanlı özellikler metin kategorizasyonuna düşük puanlı olanlara nazaran daha çok katkı sağlarlar.

2.2 Veri Seti

Web içerik sınıflandırması için makine öğrenmesi çalışmasında kullanılan veri seti, web sayfalarının 17 sınıf etiketi ile sınıflandırılmış, her etiketin veri setinin büyüklüğüne göre en az 250 en çok 1750 adet veri ile ilişkilendirildiği, en büyük veri setinin toplamda 30350 veri içerdiği bir veri kümesini temsil eder. Çalışmalar her sınıf başına 250, 1000 ve 1750 adet etiketli veri düşecek şekilde farklı boyutlardaki veri setleri ile yapılmıştır. Normalleştirme ve vektörleştirme gibi ön işlem adımlarının sınıflandırma algoritmalarının tahmin başarımları üzerindeki etkileri göz önüne

alındığında her bir sınıflandırma modeli için kullanılan eğitim ve test verisi farklılık gösterebilmektedir.

Çizelge 2.1 : Belirlenen sınıflar ve Türkçe karşılıkları

İngilizce Sınıf İsimleri	Türkçe Sınıf İsimleri
Advertising_and_Marketing	Reklam ve Pazarlama
Alcohol	Alkol
Business_and_Economy	İş ve Ekonomi
Computer_and_Internet	Bilgisayar ve İnternet
Education	Eğitim
Gambling	Kuma ve Bahis Oyunları
Games	Oyunlar
Government_and_Politics	Hükümet ve Politika
Health	Sağlık
Music	Müzik
News_and_Media	Haberler ve Medya
Religion	İnanç
Sexual_Content	Cinsel İçerik
Social_Networking	Sosyal Ağlar
Travel_and_Tourism	Seyahat ve Turizm
Video_and_Movies	Video ve Filmler
Violence	Şiddet

Veri seti oluşturulurken kategorileri belirlenmiş birçok web sitesine ihtiyaç duyulmuştur. Bu ihtiyaç doğrultusunda bir araya getirilen sınıflandırılmış web sitelerinin bir bölümü, en iyi web sitelerini sınıflandırılmış ve listeler halinde sunan Similar Web[1] ve Alexa[2] web hizmetlerinden, bir bölümü de el ile sınıflandırma çalışmaları ile elde edilmiştir. Bu web hizmetleri ve çalışmalardan temin edilen sınıflandırılmış web siteleri listelerinde yanlış sınıflandırmalar ile karşılaşılmıştır. Bu nedenle tüm bu kaynakların tek başlarına kullanılması uygun görülmemiştir. Her kaynaktan alınan veriler karşılaştırılmış ve bu sayede tüm kaynaklarda en yüksek ortaklığı gösteren veriler sisteme dahil edilmiştir. El ile sınıflandırma çalışmalarının temel amacı Türkçe web sitelerini de veri setine doğru bir şekilde eklemektir. Çünkü web hizmetleri ve yabancı çalışmalardan elde edilen sınıflandırılmış web sitesi verileri bölgesel, bilhassa Türkçe web siteleri üzerinde ciddi derecede hatalı sonuçlar içermektedir. Bunun ana sebebinin toplumlar arasındaki kültürel ve kavramsal

[1] https://www.similarweb.com/top-websites
[2] https://www.alexa.com/topsites

farklılıklar olduğu öngörülmektedir.

Çalışma kapsamında web siteleri sınıflandırılırken kullanılan 17 kategori internet servis sağlayıcıların istatistikleri ve uzman görüşleri ışığında belirlenmiş, öncelikli hedef olarak gerçek hayatta en çok ziyaret edilen web sayfalarının ortaklıkları göz önünde bulundurulmuştur.

2.2.1 Veri setinin hazırlanması

Araştırmada kullanılan veri seti web sayfalarının ham metinlerinden oluşmaktadır. Bu ham metinler web tasarımına dair birçok HTML ve CSS kodu içermektedir. Bu tür bir veri setinin dikkatlice taranmadan analizi yanıltıcı sonuçlar doğuracaktır.

Çalışma sınıflandırılmış web sitelerinin içeriklerinin bir Python betiği aracılığıyla elde edilmesi ile başlar. Boyut indirgeme, temizlik ve ön işleme süreçlerinin sınıflandırma başarımını etkileyen süreçler olması nedeniyle, web sitesi içerikleri internetten indirildikleri şekilde, özelleştirilmeden depolanmıştır. Şekil 2.2'de kısmi bir örneği verilen sınıflandırılmış web siteleri listesi betik tarafından satır satır okunarak parçalanır ve virgülden sonraki alan adam kısmı internet üzerinden sorgulanır. Sorgunun cevabı ilk olarak web sayfasının karakter kodlaması türünün (encoding) tespiti için kullanılır. Bu tespit Python'ın googletrans[3] kütüphanesini kullanılarak yapılır. Tespite ait güven katsayısı, belirlenen eşiğin üzerinde ise iddia edilen kodlama türü kabul edilir, aksi taktirde varsayılan kodlama türü olarak UTF-8 karakter kodlaması kullanılır. Ham metin çoğunlukla web sitesine ait tüm anlamlı bilgileri ve kullanıcının anlayamayacağı türden İnternet haberleşmesine ve tasarıma dair birtakım kodları içerir. Tüm bu ham metinler dosya isimlerinde web sitesi ve sınıf bilgisi içerecek şekilde metin dosyası olarak diske kaydedilir.

2.2.2 Metin temizliği

Anlamsız ve öğrenmeyi zorlaştırıcı bir takım karakter, sayı ve harflerin adım adım temizlenmesi kaliteli bir veri seti ile çalışmak için önemli bir işlemdir. İnternet üzerinden elde edilen ham metin dosyalarının temizlik aşamalarında sıklıkla, Python dilinde yazılmış doğal dil işleme kiti olan NLTK, chardet ve html2text kütüphaneleri kullanılmıştır.

[3] https://cloud.google.com/translate

Advertising_and_Marketing,ads.gaminguniverse.de	Government_and_Politics,afad.gov.tr
Advertising_and_Marketing,ads.smartclick.com	Government_and_Politics,mit.gov.tr
Advertising_and_Marketing,ads.tripod.lycos.co.uk	Government_and_Politics,saglik.gov.tr
Advertising_and_Marketing,ads.virtualcountries.com	Government_and_Politics,finance.gov.sk
Advertising_and_Marketing,adrapromosyon.com.tr	Health,aidshealth.org
Advertising_and_Marketing,hytanitim.com.tr	Health,aidshelpline.org.uk
Advertising_and_Marketing,marketingsource.com	Health,baymedicalcenter.com
Advertising_and_Marketing,marketingtoday.com	Health,ankaraftrh.saglik.gov.tr
Advertising_and_Marketing,marketingtom.com	Music,birzamanlar.net
Alcohol,wineaccessory.com	Music,neytaksim.com
Alcohol,wineanddinetour.com	Music,radios.com.br
Alcohol,wineandvoyages.com	Music,spotify.com
Alcohol,wineanorak.com	News_and_Media,accuweather.com
Alcohol,wineanswers.com	News_and_Media,atv.info
Alcohol,winearomawheel.com	News_and_Media,sporx.com
Business_and_Economy,airbus.com	News_and_Media,sportsgaming.com
Business_and_Economy,airconcoltd.com	Religion,churchinfresno.org
Business_and_Economy,aisa.org.af	Religion,churchinhollywood.com
Business_and_Economy,aitoc.com	Religion,churchinirvine.org
Business_and_Economy,cocacola.com.tr	Religion,diyanet.gov.tr
Business_and_Economy,clearlysalesjobs.co.uk	Sexual_Content,....com
Computer_and_Internet,androidinfo.net	Sexual_Content,....com
Computer_and_Internet,androidnews24.com	Sexual_Content,....com
Computer_and_Internet,asusdriver.ucoz.net	Sexual_Content,....com
Computer_and_Internet,asystems.net	Sexual_Content,....com
Computer_and_Internet,chip.com.tr	Social_Networking,arasana.com
Computer_and_Internet,chipworks.com	Social_Networking,datingxpartner.unblog.fr
Education,acibadem.edu.tr	Social_Networking,linkedin.com
Education,akdeniz.edu.tr	Social_Networking,twitter.com
Education,berkeley.edu	Social_Networking,youtube.com
Education,itu.edu.tr	Travel_and_Tourism,airchartertravel.com
Education,library.bilkent.edu.tr	Travel_and_Tourism,citycar.ee
Education,library.boun.edu.tr	Travel_and_Tourism,cityhotelmuenchen.de
Gambling,betplatinum.ag	Travel_and_Tourism,cityhotelodense.dk
Gambling,betradar.com	Travel_and_Tourism,cityislandmuseum.org
Gambling,betterbetting.com	Video_and_Movies,youtube.com
Gambling,betterbridge.com	Video_and_Movies,streamer.com
Gambling,pokerligne.org	Video_and_Movies,streaming.sunrise.ch
Gambling,pokerproxy.com	Video_and_Movies,streamops.aol.com
Gambling,pokersharkpool.com	Violence,guns.com
Games,arcadexgames.com	Violence,gunsamerica.com
Games,chess.net	Violence,hanserecords.de
Games,realmwars.blogspot.com	Violence,battletanks.com
Games,steamcommunity.com	Violence,bbguns.co.uk

Şekil 2.2 : Sınıflandırılmış web siteleri liste yapısı (kısmi örnek)

Şekil 2.3'te İstanbul Teknik Üniversitesi[4] web sitesinin Ana sayfa içeriğinin ham metin formatı yer almaktadır. Metin temizliği işlemleri sonrasında kullanıcı için anlamsız olan, Internet teknolojisinin ve mimarisinin arka planını ilgilendiren tüm kod ve

[4] www.itu.edu.tr

etiketler temizlenmiştir. Şekil 2.4, metin temizliği işleminin Şekil 2.3'te yer verilen içeriğe dair nihai çıktısını oluşturur.

Metin temizliğinde NLTK ve html2text kütüphanelerine ait fonksiyonların kullanılmasının temel amacı ham metinlerinde yer alan HTML ve CSS argümanlarında yer alan anlamlı birtakım kelimelerin de veri setine karışmasını engellemektir. Çünkü bu kelimeler neredeyse her web sitesinin ham metninde yer alan, sınıflandırma da öznitelik olarak katkısı olmayacak, üstelik verinin boyutunu ve gürültüyü artıracak kelimelerdir. Bunlara, web tasarımında nesne biçimlendirmeyi sağlayan border, layout, text vb. kelimeler örnek verilebilir.

ref="http://www.sks.itu.edu.tr/index.php?option=com_content&view=article&id=66&Itemid=43" target="_blank">Sağlık Hizmetleri Rehber Personel Portalı Yüksek Başarımlı Hesaplama Merkezi Akademisyen Siteleri Kampüste İş İmkanları Kariyer Merkezi Bilgi İşlem Sürekli Eğitim Merkezi   <h4>KAMPÜSLERDE YAŞAM</h4> Kütüphane Yemekhane Kültür Sanat Birliği Spor Tesisleri Sosyal Tesisler Kampüslerde Ulaşım Ring Servisleri Yurtlar Lojmanlar İTÜ Radyosu   <h4>BİRİMLER</h4> Genel Sekreterlik <!-- /
<a

Şekil 2.3 : İstanbul Teknik Üniversitesi Web sayfası metin belgesi içeriği (kısmi örnek)

twitter sayfası images default source twitter facebook sayfası images default source facebook instagram sayfası images default source instagram sfvrsn youtube kanalı images default source youtube sfvrsn foursquare images default source default album foursquare sfvrsn kalite ninova öğrenci işleri webmail kutuphane rehber iletişim global istanbul teknik universitesi anasayfa images default source stanbul teknik üniversitesifafbba sfvrsn hakkinda genel genel bilgiler vizyon misyon tarihçe rektorluk yonetim kurulu senato organizasyon şeması atama ölçutleri medyada sayılarla sosyal medya kurumsal kimlik hizmetler öğrenci işleri merkezi erasmus ofisi laboratuvarları yurt burs hizmetleri sağlık hizmetleri rehber personel portalı yuksek başarımlı hesaplama merkezi akademisyen siteleri kampuste imkanları kariyer merkezi bilgi işlem surekli eğitim merkezi kampüslerde yaşam kutuphane yemekhane kultur sanat birliği spor tesisleri sosyal tesisler kampuslerde ulaşım ring servisleri yurtlar lojmanlar radyosu birimler genel sekreterlik idari birimler ofisler akademik yerleşkeler ayazağa taşkışla maçka gumuşsuyu tuzla kktc fakülteler inşaat mimarlık makina elektrik elektronik maden kimya metalurji işletme gemi inşaatı deniz bilimleri edebiyat uçak uzay bilimleri denizcilik tekstil teknolojileri tasarımı bilgisayar bilişim enstitüler enerji bilimleri sosyal bilimler avrasya bilimleri bilişim deprem muhendisliği afet yonetimi öğrenim birimleri yabancı diller yuksekokulu turk musikisi devlet konservatuvarı muzik ileri araştırmalar merkezi miam uluslarası ortak lisans programları insan toplum bilimleri bolumu guzel sanatlar bolumu ataturk ilkeleri inkılap tarihi bolumu beden eğitimi bolumu turk dili bolumu bolumler akreditasyon akredite lisans programları araştirma araştirma araştırma merkezleri laboratuvar altyapı bilgi sistemi araştırma

Şekil 2.4 : Metin temizleme süreci ara çıktısı (kısmi örnek)

2.3 Veri Önişleme

Veri önişleme makine öğrenmesi ve veri madenciliğinde kullanılan önemli bir adımdır. Bu çalışmada kullanılacak olan veri seti yüksek boyutlarda olduğundan veri önişleme süreci son derece kritik bir role sahiptir. Öznitelik vektörü, metin temsili öznitelikler olarak seçildiğinde kelime frekansına göre ağırlıklandırılır.

Alakasız bilgilerin ve gürültünün çokluğu eğitim aşamasında bilgi keşfini zorlaştırır. Bu aşamada veri ön işleme temizlik işlemlerini içerir. Normalizasyon, boyut indirgeme, kök çözümleme vb. yöntemlerin çıktısı nihai veri setini oluşturacaktır.

Verinin yüksek boyutlara ulaşması ve bu verilerin sadeleştirilmeden bir model kurmada kullanılması modelin eğitilmesi için ihtiyaç duyulan zaman ve kaynak miktarını ciddi ölçüde arttırmaktadır. Çizelge 2.1'de veri seti üzerindeki önişleme süreçlerinin sınıflandırma algoritmalarının model eğitim süreleri üzerindeki etkileri gösterilmiştir. Örnek gösterim için lojistik regresyon sınıflandırma algoritması ile eğitilmiş bir modelin hesaplama süresi baz alınmıştır.

Çizelge 2.2 : Veri seti üzerindeki önişleme süreçlerinin sınıflandırma algoritmalarının model eğitim süreleri üzerindeki etkileri

Veri Seti	Donanımı	Önişlemesiz veri seti işlem süresi	Önişlemeli veri seti işlem süresi
Küçük boyutlu veri seti	4 çekirdekIntel Core i5 2.6 Ghz	00:17.35	00:03:56
Büyük boyutlu veri seti	40 çekirdek Intel Xeon Gold 2,4 Ghz	05:27:16	01:56:41

Metin içeriği ile ilgili bilgi sahibi olmayı sağlayan önemli öznitelikler yerine anlamsız ve çok sık tekrarlanan karakter ve harf grupları ön plana çıkmaktadır. Sınıflandırma algoritmalarının kullandığı parametreler ile maksimum öznitelik sayısını sınırlandırmakta, ancak bir önceki cümlede ifade edilen karakterlerin çokluğu nedeniyle bu işlem anlamlı kelimeleri kapsam dışı bırakmaktadır. Ayrıca yüksek boyutlu verilerin görselleştirilmesi de bir o kadar zordur.

2.3.1 Boyut indirgeme (Dimensionality reduction)

Veri bilimi ve makine öğrenmesi çalışmalarında kullanılan veri setleri yüksek boyutlu diğer bir deyişle çok fazla özniteliğe sahip olmaktadır. Verinin yüksek boyutlara ulaşması ve bu verilerin temizlenmeden bir model kurmada kullanılması ihtiyaç duyulan zaman ve kaynak miktarını ciddi ölçüde arttırmaktadır. Ayrıca yüksek boyutlu verilerin görselleştirilmesi de bir o kadar zordur. Yapılan birçok çalışmada kullanılan metodoloji ve ortaya çıkan sonuçlar ancak görsel argümanlarla sunulduğunda insanlara çok daha anlaşılır şekilde aktarılabilmektedir. Yüksek boyutlu verilerin görselleştirmeyi zorlaştırmanın yanı sıra bazı öznitelikleri arasında yüksek korelasyona sebep olması da önemli bir sorundur. Böyle bir veri seti gereksiz bilgi

ihtiva eder ve bu verilerle kurulan modelde aşırı öğrenme(overfitting) gibi sorunlara sebep olabilir.

Boyut azaltmada kullanılan birçok yaklaşım vardır. Bu çalışmada normalizasyon, Stemizasyon ve Lemmatizasyon yöntemleriyle boyut azaltma gerçekleştirilmiştir.

2.3.1.1 Dil analizi

En çok ziyaret edilen web sitelerinin ana sayfaları yaklaşık %80 oranında İngilizce dilindedir. Rusça, İspanyolca ve Almanca başta olmak üzere diğer birçok dilde değişen miktarlarda bilgi içeren web sayfası bulunmaktadır. Türkçe de %2'lik oranla en çok kullanılan web içeriğine sahip 9'uncu dildir.

Çalışma kapsamında gerçekleştirilen dil bazlı analizlerde kullanılmak üzere İngilizce, Almanca, Rusça, İspanyolca, Fransızca, Türkçe ve Portekizce dilleri değerlendirmeye alınmıştır. Bu dillerdeki her bir web sayfası metni üzerinde kök çözümleme yöntemleri ve morfolojik analizler ile çeviri yöntemi farklı sıralar ile uygulanmış ve sınıflandırma başarısı üzerindeki etkileri incelenmiştir. Araştırma sırasında belirlenen 7 dil dışında kalan dillere sahip metinler veri setinden ayrılır.

Çalışma kapsamında, bir web sayfasına ait anlamlı metinlerin depolanması ile meydana gelen metin dosyalarının boyutu dikkate alınarak, en çok anlamlı içeriğe sahip metinler seçilmiştir. Metinler üzerinde gerçekleştirilen kök çözümleme ve morfolojik analiz işlemleri için metinlerin hangi dillerde yazılmış olduğu bilgisine ihtiyaç duyulur. Dil saptama işlemi için iki farklı yöntem kullanılmış ve başarı oranları analiz edilerek karşılaştırılmış.

Bu yöntemlerin ilki Google[5] şirketinin makine öğrenimini kullanarak diller arasında dinamik çeviriye olanak sağlayan Translation API[6] uygulama programlama ara yüzünün kullanılmasıdır. Önişleme sürecinin sonraki aşamalarındaki çeviri işlemlerinde, bu servis kullanılarak Python dili için hazırlanan googletrans [7]uygulama programlama ara yüzü kullanılmıştır. Bu ara yüz bir metnin dilini tahmin etmenin yanında tahminin başarı oranını ifade eden bir güven katsayısı sağlar. Çalışma kapsamında 500 web sayfası metni üzerinde yapılan dil tahminlerinde algoritmanın

[5] https://www.google.com/
[6] https://cloud.google.com/translate
[7] https://cloud.google.com/translate

476 metnin ağırlıklı dili hakkında doğru bilgi verdiği ortaya çıkmıştır. Bu doğru tahminlerdeki güven katsayısının %70 ve üzeri olduğu tespit edilmiştir.

İkinci bir yöntem olarak etkisiz kelime skorlama ile dil tespiti yapan bir yaklaşım gerçekleştirilmiştir. Bu yaklaşımda bir metinden belirlenen bir dizi dile ait etkisiz kelimeler çıkarılır. Çıkarılan kelime sayısı dil etiketiyle birlikte raporlanır. En yüksek değere sahip dil ilgili metnin dilidir denilir. Bu yaklaşımın 500 web sayfası metni üzerinde 439 metnin ağırlıklı dili hakkında doğru bilgi verdiği ortaya çıkmıştır.

Metin dili tespitinde yukarıda bahsedilen iki yöntem birlikte kullanılmıştır. Dil tespiti yapan ilk yöntemin %70'in altında bir güven katsayısına sahip olduğu durumlarda metnin dilini ikinci yöntem belirlemektedir. Aksi taktirde ilk yöntem dil bilgisini tayin etmektedir.

2.3.1.2 Normalleştirme

Büyük veri setleri üzerinde veriyi normalize etmeden herhangi bir algoritma kullanılırsa, ölçekleme sorunları nedeniyle vektörleri birleştirmek zor olur. Normalleştirme yakınsama için verileri daha iyi şartlandırır. Normalleştirme, bir yakınsama probleminin büyük bir varyansa sahip olmamasını ve optimizasyonun mümkün olmasını sağlar. [11]

En basit ifadesi ile normalleştirme verileri 0 ile 1 arasında değerlere sahip olacak şekilde yeniden ölçeklendirmektir.

2.3.1.3 Gövdeleme (Stemming)

Metin bazlı sınıflandırmanın ana bileşeni metni oluşturan kelimelerdir. Ancak bu kelimeler çoğu zaman aynı köke sahip olsalar dahi aldıkları eklerden dolayı öznitelik çıkarımı adımında farklı birer kelime olarak değerlendirilmektedir [28]. Stemming algoritmaları da bir kelimede bulunabilecek tüm ekleri ve dönüşümleri tanımlı ek listelerini göz önüne alarak belirler ve kelimelerin anlamlı en temel köklerini bulmayı hedefler. Çalışmanın boyut indirgeme basamaklarında Snowball Stemmer algoritması kullanılmıştır. Normal veri seti ve Snowball Stemmer algoritmasının kullanıldığı veri setinin örnek karşılaştırılmasına Şekil 2.5'de yer verilmiştir.

2.3.1.4 Anlamsal olarak köke inme (Lemmatization)

Lematizasyon kelimelerin morfolojik analizini dikkate almaktadır. Bu nedenle

metodolojinin temel mantığı dilbilime dayanır. Burada her kelimenin lemma'sı çıkartılmaktadır. Ancak her kelimeye ait doğru lemma'nın çıkartılabilmesi her kelimenin morfolojik analize bakmakla mümkündür.

Bir web sayfasına ait metin içeriğinin kelimelerine ayrılmış formu	home category body type treatment diet massage weight loss search medicine system natural healing becomes among conscious citizen world rooted count year existence country long time unchanged keeping secret original system today popular find follower world importance look traditional even medicine concept firstly system translate science life life science system
Aynı metnin Snowball Stemmer ile yeniden işlenmiş görünümü	home categori bodi type treatment diet massag weight loss search medicin system natur heal becom among conscious citizen world root count year exist countri long time unchang keep secret origin system today popular find follow world import look tradit even medicin concept first system translat scienc life life scienc system

Şekil 2.5 : Snowball Stemmer algoritmasının kelime torbası üzerinde uygulanması

Bu durum da algoritmanın biçimi lemma'ya bağlayarak inceleyebileceği detaylı sözlüklere sahip olması gerekir. Çalışmanın boyut indirgeme aşamalarında veri seti, Lematizasyon algoritmaları ile sadeleştirilerek yeni bir veri seti oluşturulmuştur. Bu veri seti sınıflandırma algoritmalarının her birine girdi oluşturarak inşa edilen modellerin başarımları karşılaştırılarak sonuç kısmında sunulmuştur.

Bir web sayfasına ait metin içeriğinin kelimelerine ayrılmış formu	Değişim sürecinden başarılı çıkanlar çelişkileri kavgaya dönüştürüp galip gelenler değil, bu değişimi herkesin yoluna uygun şekilde yönetenler oluyor. Toplumsal değişimi yönetmek için gereken tek sermaye ise bilgi. Bilgi farklı yöntemlerle tedarik edilen veriler yığını olmamalıdır. Aslında bilgi bir yandan gerçeğin yapılandırılmasıdır. Bilgi, güvenilir, gerçek, tarafsız, anlamlı ve anlaşılır olmalıdır. Son gelişmelerden haberdar olmak için aylık bültenimize abone olun.
Aynı metnin Lemmatization ile yeniden işlenmiş görünümü	değişim sürecinden başarılı çıkanlar çelişkileri kavgaya dönüştürüp galip gelenl değil, bu değişimi herkesin yoluna uygun şekild yönetenl oluyor. toplums değişimi yönetmek için gereken tek sermay ise bilgi . Bilgi farklı yöntemlerl tedarik edilen veril yığını olmamalıdır. Aslında bilgi bir yandan gerçeğin yapılandırılmasıdır. Bilgi, güvenilir, gerçek, tarafsız, anlamlı ve anlaşılır olmalıdır. Son gelişmelerden haberdar olmak için aylık bültenim abon olun.

Şekil 2.6 : Lemmatization algoritmasının kelime torbası üzerinde uygulanması

Gövdeleme ve anlamsal olarak köke inme arasındaki en belirgin fark, gövdeleme işleminin bir kelimeyi doğruca köküne indirirken anlamsal olarak köke inme işleminde kelimenin anlamı dikkate alınarak köküne inme sağlanmış olur. [27]

Normal veri seti ve Snowball Stemmer algoritmasının kullanıldığı veri setinin örnek karşılaştırılmasına Şekil 2.6'da yer verilmiştir.

2.3.1 Önerilen önişleme modeli

Anlamsız ve öğrenmeyi zorlaştırıcı bir takım karakter, sayı ve harflerin adım adım temizlenmesi kaliteli bir veri seti ile çalışmak için önemli bir işlemdir. İnternet üzerinden elde edilen ham metin dosyalarının temizlik aşamalarında sıklıkla, Python dilinde yazılmış doğal dil işleme kiti olan, Stemmer ve Lemmatizer fonksiyonlarını sağlayan nltk[8] kütüphanesi, çeviri ve dil analizi sağlayan googletrans, Python temel işlem kütüphanesi Pandas[9] ve HTML teknik içeriğinin metinden temizlenmesini sağlayan html2text[10] kütüphaneleri kullanılmıştır.

Bir web sitesine ait ham metin üzerinde uygulanan önişleme süreçleri 3 farklı yaklaşımla gerçekleştirilmiş, tüm süreçlerin makine öğrenmesi başarımına etkileri gözlemlenmiştir. Bu yaklaşımlar Şekil 2.7'de ortak ve farklı adımları ile birlikte gösterilmiştir.

Deneysel çalışmanın gerçekleştirildiği her yöntem için ortak koşulan adımlarda:

- HTML-anlamlı metin dönüşümü, tüm HTML ve CSS etiket ve kodlarını metinden çıkartır.
- Dil analizi bölümünde açıklanan yöntemler kullanılarak metnin ağırlıklı dili tespit edilir. Tespit edilen dil belirlenen güven yüzdesini sağlıyorsa metnin dili olarak kabul edilir. Eğer dil tespiti güvenilmez kabul edilen %70 altında bir sonuç verirse metin dili çalışma kapsamında önerilen etkisiz kelimelerin skorlanması yöntemi ile belirlenir.

YÖNTEM 1

- Bu yöntemde herhangi bir çeviri işlemi uygulanmaz.
- Metinde yer alan tüm anlamsız karakterler ve çoğul boşluklar temizlenir.

[8] https://pypi.org/project/nltk/
[9] https://pypi.org/project/Pandas/
[10] https://pypi.org/project/html2text/

- Etkisiz kelimeler metinden çıkartılır.
- Dil tespit aşamasında belirlenen dil bilgisi kullanılarak dil bazlı gövdeleme ve anlamsal olarak köke inme işlemleri uygulanır.
- Nihai çıktı Kategori,Web sitesi,Dil,İçerik formatında diske kaydedilir.

Çeviri yapılmak istenen bir yöntem seçildiğinde metin dili, belirlenen 7 dil içerisinde yer almadığı durumlarda metin etkisiz kelimelerin dil bazlı silinmesi adımından itibaren Yöntem 1'e dahil edilir. Metin dili, belirlenen 7 dilden birine sahip olduğunda Yöntem 2 ve Yöntem 3'ün hazırlık aşaması olan sınırlı karakter temizliği işlemi gerçekleştirilir. Burada nokta, virgül, tek tırnak ve noktalı virgül noktalama işaretleri korunarak sonraki aşamalarda yapılacak çeviri işlemi öncesinde anlam kaybı en az seviyeye indirilmeye çalışılmaktadır.

YÖNTEM 2

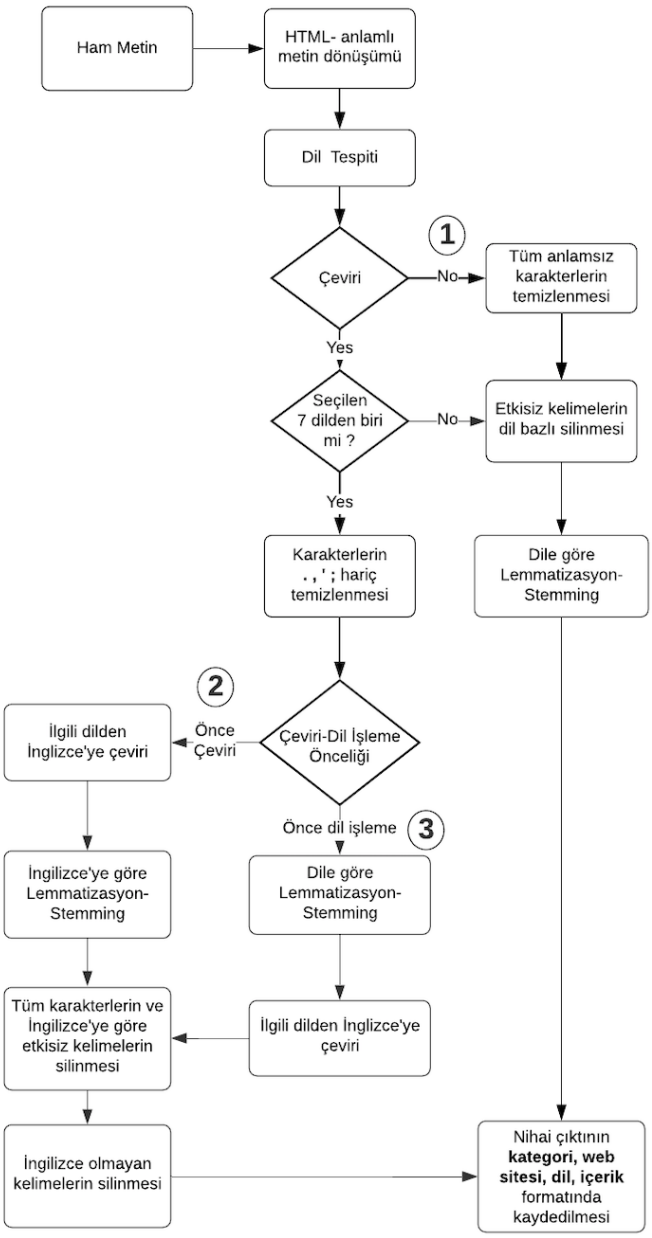
- Metin önceden belirlenen içerik dili kaynak dil olarak kabul edilerek İngilizce diline çevrilir.
- Gövdeleme ve anlamsal olarak köke inme işlemleri İngilizce dili özelinde uygulanır.
- Tüm noktalama işaretleri ve etkisiz kelimeler çıkartılır.
- İngilizce olmayan kelimeler çıkartılır.
- Nihai çıktı Kategori,Web sitesi,Dil,İçerik formatında diske kaydedilir.

YÖNTEM 3

- Metin önceden belirlenen içerik diline göre gövdeleme ve anlamsal olarak köke inme işlemlerine tabi tutulur.
- Metin içerik dili kaynak dil olarak kabul edilerek İngilizce diline çevrilir.
- Tüm noktalama işaretleri ve etkisiz kelimeler çıkartılır.
- İngilizce olmayan kelimeler çıkartılır.
- Nihai çıktı Kategori,Web sitesi,Dil,İçerik formatında diske kaydedilir.

2.3.2 Kelime vektörleştirme

Ham metinler bilgisayarlar tarafından anlaşılamazlar. Metinlerin bilgisayar dilinde bir anlam ifade etmesi ancak sayısal bir temsil ile mümkün olur. Makine öğrenmesi, istatistiksel veriler ve algoritmalarla ilgilenen bir alandır ve bu nedenle, metinlerin sayılara dönüştürülmesi gerekmektedir.



Şekil 2.7 : Önişleme süreçleri akış diyagramı

Ham metinleri sayısal formla temsil etmek için farklı yaklaşımlar mevcuttur. Bu çalışmada metinlerin sayılara dönüştürülmesi için kullanılan Kelime Çantası Modeli ve TFIDF vektörleştirme en yaygın yaklaşımlardır.

2.3.2.1 Kelimelerin torbası (Bag of words)

Metin belgelerini sayısal özelliklere dönüştürmek için Kelime Torbası Modeli kullanır. Böylece metin, kelimelerden oluşan bir çanta halini alır ve her bir kelime ayrı bir nesne olarak incelenir. Kelimelerin karşılaşılma sıklıkları dikkate alınırken sıra, grammer ve anlam gibi özellikler göz ardı edilir.

Sklearn.feature_extraction modülü, ham metin ve görüntülerden özellikler çıkarmak için yöntemler içeren bir Scikit-Learn kütüphanesi alt modülüdür. Vektörleştirmenin bu aşamasında bu modülün CountVectorizer sınıfı kullanılmıştır. Bu sınıf kullanılırken belirtilmesi gereken bazı önemli parametreler vardır. İlki max_features parametresidir. Bu parametre sınıflandırıcının eğitilmesi için özellik olarak en çok kaç adet kelimenin kullanılmak isteneceğini belirtir. İkinci parametre min_df'tir. Bu, bir önceki parametrede belirtilen sayıdaki özellikleri içermesi gereken minimum doküman sayısına karşılık gelir. Benzer şekilde, max_df için de değer belirlenir; kesir yüzdeye karşılık gelir. Örneğin 0,7, tüm belgelerin en fazla %70'inde karşılaşılan sözcükleri dahil etmemiz gerektiği anlamına gelir. Bunun nedeni bir belgede bulunan her kelimenin sınıflandırma için uygun olmaması, belge hakkında bir bilgi vermemesidir. Son olarak, stop_words parametresi ile her dilde, dile özgü olan, 2 veya 3 harften oluşan edat ve bağlaçlar metinden kaldırılır, çünkü duyarlılık analizi yapıldığında, bu sözcükleri herhangi bir yararlı bilgi içermeyecektir. Bu parametrenin kullanımı için nltk.corpus kütüphanesinin stopwords nesnesi parametreye aktarılmaktadır. Tüm bu parametrelerin optimizasyonu ızgara araması yöntemi ile gerçekleştirilmiştir ve bu değerlere karşılaştırma tablolarında yer verilecektir.

Vektörleştirme işleminin hedefi olan metin belgelerinin sayısal karşılıklarına dönüştürülmesi işlemi CountVectorizer sınıfının fit_transform işlevi ile yapılır.

2.3.2.2 TFIDF vektörleştirme

Kelime torbası yaklaşımı, metnin sayılarla temsil edilmesinde başarılır bir yöntemdir. Ancak, önemli bir dezavantajı vardır. Bu yaklaşım belgede oluş sıklığına bağlı olarak bir kelimeye bir puan atar. Sözcüğün başka belgelerde de yüksek oranda ortaya çıkma

olasılığı olduğu dikkate alınmaz. TFI-DF, bu sorunu bir kelimenin terim sıklığını ters belge sıklığı ile çarparak çözer. TF, "Terim Frekansı" anlamına gelir. Bir terimin, IDF "Ters Belge Frekansı" anlamına gelir.

$$w_{i,j} = tf_i \; x \log\left(\frac{N}{df_i}\right) \quad \textbf{(2.1)}$$

Yukarıdaki formülde (2.1) değeri herhangi bir terimin belgede bulunma sayısının, belgede en sık bulunan terimin bulunma sayısına oranıdır. N/ ise toplam belge sayısının ilgili terimin bulunduğu belge sayısına oranıdır.

Belirli bir belgedeki bir kelimenin için, söz konusu kelimenin görülme sıklığı bir belgede daha yüksek ancak diğer belgelerde daha düşük ise o kelimenin TFIDF daha yüksektir. Çalışmanın vektörleştirme adımlarında ayrı ayrı hem Kelime Torbası hem de TFIDF metodu ayrı ayrı uygulanmış, Rassal Orman sınıflandırma modeli haricindeki tüm modellerde Bag of words metodu daha yüksek sonuçlar vermiştir.

3. SINIFLANDIRMA MODELİ

İstatiksel sınıflandırmada sınıflandırma modelinin inşası iki adımdan oluşan bir süreçtir. İlk olarak her verinin bir sınıfına ait olduğu kabulüyle bir veri seti temin edilir ve bu veri setinin bir kısmı eğitim veri seti olarak kullanılır. Sonraki adımda ise seçilen bir sınıflandırma algoritması ile sınıfları bilinmeyen verilerden oluşan bir test veri seti üzerinde tahminler yapılır. Tahminlerden elde edilen sonuçlar ile bilinen sınıf etiketleri karşılaştırılır. Bu karşılaştırmanın sonucu modelin doğruluk derecesini gösterir. Bu doğruluk derecesi başarılı kabul edilirse model artık yeni verilerin sınıflandırılmasında kullanılabilir.

Etiketlenmiş bir veri setinde öznitelik vektörünün X, etiketlerin Y ve sınıflandırma modelinin C olduğu kabulüyle; işlev C(X) fonksiyonu tarafından Y değerlerinin tahminidir. Temel fikir, X'in C altındaki sınıflara ait olma olasılıklarının tahmin edilmesidir.

Yeni verilerin sınıflandırma tahmin sonuçları çoklu sınıflı ve çoklu etiketli olmak üzere iki tür sınıflandırma prensibi ile değerlendirilir. Çoklu sınıflı sınıflandırmada her veri herhangi bir sınıfa ait olabilir ancak aynı anda yalnız tek bir sınıfa atanabilir. Çoklu etiketli sınıflandırmada ise bir verinin aynı anda birden fazla sınıfa ait olabileceği varsayımı yapılır.

Bu çalışmada web sayfalarının anlamlı metinlerinden oluşturulmuş olan her bir veri tek bir sınıf ile ilişkilendirilmiştir. Bu nedenle temel prensip çoklu sınıflı sınıflandırma modeline uygundur. [13]

3.1 Kullanılan Sınıflandırma Algoritmaları

Bu çalışmada aşağıdaki sınıflama ve regresyon modelleri kullanılmıştır:

- Lojistik Regresyon
- Destek Vektör Makineleri (SVM)
- Karar Ağaçları (Decision Tree)

- Rassal Orman (Random Forest)
- Gaussian Naive-Bayes
- KNeighbors Classifier
- OneVSRest Classification
- Ensemble Classification

3.1.1 Logistic regression sınıflandırma modeli

Logistik Regresyon tahmin edilmek istenilen değişkenin kategorik bir değişken olduğu durumlarda kullanılabilen bir sınıflandırma modelidir. Tüm regresyon analizleri gibi, lojistik regresyon da öngörücü bir analizdir. Verileri tanımlamak ve bir bağımlı ikili değişken ile bir veya daha fazla bağımsız değişkenler arasındaki ilişkiyi açıklamak için lojistik regresyon kullanılır. Diğer bir ifade ile, y değişkenin olma olasılığı ile veri setindeki x öznitelikleri arasındaki ilişkinin anlaşılmasını sağlar. Giriş değerleri (x), bir çıkış değerini (y) tahmin etmek için ağırlıklar veya katsayı değerleri kullanılarak doğrusal olarak birleştirilir. Lojistik regresyonun ikili lojistik regresyon, multinomiyal lojistik regresyon ve sıralı lojistik regresyon olmak üzere 3 tipi vardır. Kategorik cevabın yalnızca iki sonuca sahip olduğu durumlarda ikili lojistik regresyon (Binary Logistic Regression) kullanılır. Belirli bir sıraya tabi olmayan, 3 veya daha fazla kategorik cevabın bulunduğu durumlar için multinomiyal lojistik regresyon (Multinomial Logistic Regression) kullanılır. Kategorik sınıflandırmanın belirli bir aralıkta sıralı sonuçlar verdiği durumlar için ise Sıralı Lojistik Regresyon (Ordinal Logistic Regression) kullanılır. Bu çalışmada sıralı olmayan 17 farklı kategorik cevap bulunacağından multinomiyal lojistik regresyon kullanılmıştır.

Lineer Regresyon modeli regresyon için başarılı olsa dahi sınıflandırma için başarısızdır. Bunun nedeni doğrusal bir modelin olasılıkları vermemesidir. Doğrusal regresyon iki sınıf olan durumlarda sınıflardan birini 0, diğerini 1 ile etiketler ve bu sınıfları sayılar olarak görür. Daha sonra noktalar ve hiper düzlem arasındaki mesafeleri en aza indirir. Ancak tahmin edilen sonuç bir olasılık değil noktaların enterpolasyonundan ibarettir. Bu da bir sınıfın diğerinden ayrılması için anlamlı bir eşiğin olmadığı anlamına gelir.

Lineer Regresyon yönteminde bağımsız değişkenlerin parametreleri Ordinary Least Squares (OLS) (Carl Friedrich Gauss, 1795) yöntemiyle tahmin edilirken Lojistik

regresyonda parametreler Maximum Likelihood (MLE) (R. A. Fisher, 1912) yöntemiyle hesaplanır. Sıradan En Küçük Kareler (OLS) regresyonu, bir veya daha fazla bağımsız değişken ve bir bağımlı değişken arasındaki ilişkiyi tahmin eden istatistiksel bir analiz yöntemidir [14]. Yönteme ismini veren bu en küçük kareler veri kümesinden elde edilen yanıtlar ile doğrusal yaklaşımla öngörülen yanıtlar arasındaki kare dikey mesafelerin toplamını en aza indirerek elde edilir. Özetle doğrusal bir regresyon modelinde bulunan bilinmeyen parametreleri yaklaşık olarak belirlemek için bir yöntemdir.

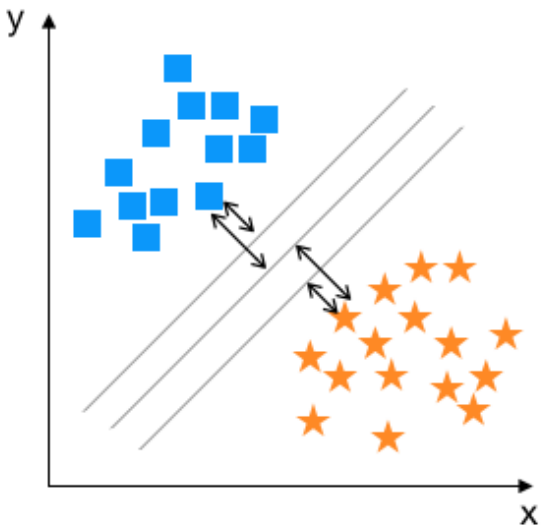
Lojistik regresyonda kullanılan Maksimum Olabilirlik Yaklaşımı (MLE) yöntemi ise çeşitli makine öğrenme algoritmaları tarafından kullanılan ortak bir öğrenme algoritmasıdır ve görevi sonsuz sayıdaki parametre içinden veri setinin görülme olasılığını maksimize eden en iyi parametreleri seçmektir. Bir diğer deyişle istatistiksel bir modelin parametrelerinin tahmininde ve bu modelin veri kümesindeki verilere uydurulmasında kullanılır [15]. Veri setinde gözlemlenen normal dağılımın ortalama değeri modelin öngörüsü ile uyuşmayabilir. Buradaki ortalama değerin verinin değil dağılımın ortalama değeri olduğu unutulmamalıdır. Bu uyuşma en basit ifadesiyle modelin normal dağılımın ortalama değerinin olduğu bölgeye kaydırılması olarak da açıklanabilir. Öngörülen değerlerin veri setinde görülme olasılığını artırmanın yolu belirlenecek uygun parametreler sayesinde veri seti dağılım ortalama değeri ile öngörü sıklığının olduğu bölgeleri örtüştürmektir. Bu sayede öngörülen değerlerin veri setinde gözlemlenme olasılığı maksimuma çıkmış olacaktır. Özetle Maksimum Olabilirlik Yaklaşımı belirlediği parametreler ile hangi eğrinin veri setine uyma olasılığının en yüksek olduğunu söylemektedir.

3.1.2 Destek vektör makineleri sınıflandırma modeli

Destek Vektör Makinası (Support Vector Machine)[11] hem sınıflandırma hem de regresyon problemlerinde kullanılabilen denetimli bir makine öğrenme algoritmasıdır. Sınıflandırma problemlerinde daha sık kullanılmakta olan bu algoritmada, her bir veri öğesi n-boyutlu uzayda (n öznitelik sayısını belirtir) bir nokta ile temsil edilir. Farklı sınıflara ait öğelerin mümkün olan en geniş aralıklarla bölünmesi için iki sınıfı en iyi

[11] (V. Vladimir N., C. Alexey Ya., 1963)

ayıran hiper düzlem çizilir ve sınıflandırma işlemi yapılır. Diğer bir deyişle Destek Vektör Makinası iki sınıfı birbirinden ayıran sınırdır denilebilir. [16]

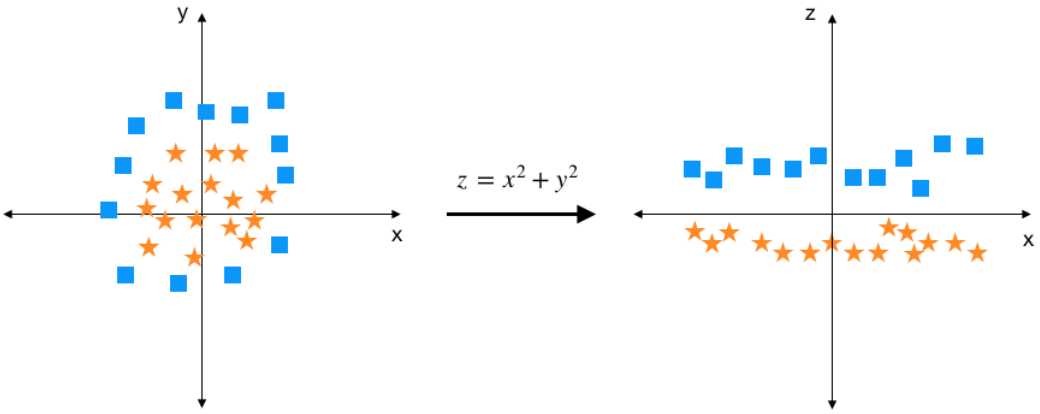


Şekil 3.1 : Destek Vektör Makinesi hiper düzlem ayrımı

İki sınıfı bir hiper düzlemle ayırmak diğer sınıflandırma algoritmalarında da kullanılan bir işlemdir. Ancak burada temel nokta doğru bir hiper düzlemin nasıl çizileceğidir. Destek vektör Makinası algoritmasında en yakın veri noktası (sınıf) ile hiper düzlemin uzaklığının maksimum olduğu düzlem en başarılı hiper düzlem kabul edilir. Burada algoritmaya ismini veren destek vektörleri hiper düzleme daha yakın olan ve hiper düzlemin konumunu belirleyen veri noktalarıdır. Bir hiper düzlemin sınıfları birbirinden doğru şekilde ayırmış olması zorunluluğu düzlem-sınıf uzaklığı maksimizasyonundan daha öncelikli bir kıstastır. Bu nedenle Destek Vektör Makinası algoritması doğru sınıflandırmayı yapan hiper düzlemi seçer. Ancak bir sınıfa ait öğenin belirgin şekilde diğer bir sınıfa ait alanın içinde yer aldığı durumlarda düz bir çizgi çizilemez. Böyle durumlarda Destek Vektör Makinası algoritması bu ve benzeri aykırı noktaları yok sayma ve maksimum marjlı hiper düzlem bulma özelliğine sahiptir.

İki sınıf arasında doğrusal bir hiper düzlemin bulunamayacağı durumlar da vardır. Böyle durumlarda Destek Vektör Makinası algoritması bir takım veri dönüşümleri yaparak ayrılamaz olan problemi ayrılabilir hale getirebilir. Örneğin iki farklı sınıfın kare ve yıldızlardan oluştuğunu varsayalım. Kare ve yıldız dağılım x ve y orijinlerinin etrafında konumlanmış olsun. Karelerin koordinat düzleminde orijine yıldızlardan daha yakın olduğu kabul edilsin. Yani bu iki sınıf birbirinden ancak bir eliptik düzlem

ile ayırabilir olsun. Burada $z = x^2 + y^2$ şeklinde bir dönüşüm yapıldığında z eksenindeki tüm değerler pozitif olacağından ve varsayımdan dolayı karelerin orijine daha yakın, yani mutlak değerleri yıldızlarınkinden daha küçük, olduğundan problem doğrusal bir hiper düzlem çizilebilir hale gelecektir. Bu işlem destek vektör makinası algoritmasında bulunan çekirdek numarası (kernel trick) tekniği [17] ile gerçekleştirilir. Bu teknikle algoritma bazı karmaşık veri dönüşümleri yapar ve tarafımızdan tanımlanan etiketlere göre verileri ayırma işlemini öğrenir. Şekil 3.2'de uzayındaki dağılımlarının doğrusal bir hiper düzleme olanak vermediği kare ve yıldız sınıfı verileri $z = x^2 + y^2$ dönüşümü ile R^3 uzayında, doğrusal bir hiper düzlem ile ayrılabilir hale gelmiştir.



Şekil 3.2 : Çekirdek numarası dönüşüm tekniği

3.1.3 Karar ağaçları sınıflandırma modeli

Makine öğrenmesi modelleri arasında çalışma mantığı insan zihnine en yakın olduğu düşünülen model karar ağaçlarıdır. İnsanların zihinlerindeki karar alma mekanizması 1-2 veya daha çok katmanlı karar ağacı derinliğine sahip olabilmektedir. Sayısal bir değerin belirli bir aralıkta olup olmadığının mukayesesi dahi insan zihnindeki en basit karar ağacı modelinin örneğidir. Karar Ağacı, anlamak ve yorumlamak için en kolay ve popüler sınıflandırma algoritmalarından biridir. Hem sınıflandırma hem de regresyon türü bir problem için kullanılabilir. Eğitim süresi, Neurol Network algoritmasına göre daha hızlıdır ve yüksek boyutlu verileri doğru bir şekilde işleyebilir.

Sınıflandırma ağacı (classification tree) veya tahmin ağacı gibi uygulamalara karar ağacı modelinin alt yöntemleri denebilir. Sınıflandırma ağacında kullanılan birçok algoritma mevcuttur. ID3, C4.5, C5.0, CART, CHAID ve QUEST bunlardan bir

kaçıdır. Bu algoritmaların başında ID3 ve C4.5 algoritmaları gelmektedir. C4.5 algoritması ID3 algoritmasının [6] geliştirilmiş halidir. Aynı şekilde C5.0 algoritması da C4.5 algoritmasının geliştirilmiş hali olup, iki algoritmadan elde edilen sonuçlar aynı olsa dahi C5.0 biçim olarak daha düzgün karar ağaçları elde etmeyi sağlamaktadır ve büyük veri setleri için kullanılmaktadır.

ID3 algoritmasında tek tek özellik vektörleri incelenir ve en yüksek bilgi kazancına sahip özellik, ağaçta dallanma yapmak için seçilir. C4.5 ağacının ID3 ağacından en büyük farkı normalleştirme kullanmasıdır. Her özelliğin normalleştirilmiş bilgi kazanımı hesaplanır. Yani ID3 ağacı üzerinde entropi veya bilgi kazancı hesabı yapılır ve bu değere göre karar noktaları belirlenir. C4.5 ağacında ise entropi değerleri birer oran olarak tutulur. Ayrıca C4.5 algoritmasında ağaç dalları kullanım sıklıklarına göre ağaç üzerinde konum değiştirebilirler. Bu da ID3 ağacının yaklaşımından farklı olarak C4.5 ağacında budama (prunning) işlemi yapılmasıdır. [18]

Karar ağaçları tekniğinde verinin sınıflandırılması öğrenme ve sınıflandırma olmak üzere iki aşamadan oluşur. Öğrenme aşamasında önceden etiketlenmiş olan veri algoritma tarafından analiz edilir ve model verilen eğitim verilerine dayanarak geliştirilir. Sınıflandırma diğer bir deyişle tahmin adımda, verilen verinin cevabını tahmin etmek için model kullanılır. Öğrenilen bu model karar ağacı olarak gösterilir. Sınıflandırma aşamasında ise model bir takım test verisi ile sınanır. Bu işlem karar ağaçlarının doğruluğunu belirlemek için gerçekleştirilir. Eğer doğruluk belirlenen eşiklerin üzerinde ise artık model yeni verilerin sınıflandırılması amacıyla kullanılabilir. [19]

Bir karar ağacı, çok sayıda kayıt içeren bir veri kümesini, bir dizi karar kuralları uygulayarak daha küçük kümelere bölmek için kullanılan bir yapıdır. Yani basit karar verme adımları uygulanarak, büyük miktarlardaki veriyi, çok küçük veri gruplarına bölerek kullanılan bir yapıdır. [20]

Karar ağaçları modelinde temel mantık bilgi kazanımının maksimize edilmesidir. Bunun sağlanması için rasgelme oranlarının minimuma indirilmesi gerekmektedir. Bu nedenle model her düğümde hata fonksiyonunu tekrar hesaplayarak en düşük hataya sahip durumu seçer. Burada amaç, gruplara ayrılan veri seti örneklerinin aynı sınıf etiketine sahip olana dek tekrar tekrar gruplara ayrılmasıdır. Bu işlem sonucunda eğitim verisinde hangi alanların hangi sıra ile ağacı oluşturacağı belirlenmiş olur.

Karar ağacı modelinde veri setinin en ayırt edici niteliği belirlenir. Bu değer karar ağacının kökü olarak alınır. Bu kök niteliğin alt düğümleri de aynı işleme tabi tutularak alt veri kümesi oluşturulur. Oluşturulan her bir alt veri kümesi için yeni en ayırt edici nitelik belirlenir ve bu nitelik bir sonraki alt veri kümeleri için kök düğümü temsil eder. Burada en ayırt edici nitelik belirlenirken bilgi kazancı ölçülür. Veri bir özelliğe göre bölündüğünde elde edilen her bir veri kümesinin belirsizliği minimum ve bilgi kazancı maksimum olmalıdır. [20]

Bilgi kazancı ölçülürken Entropy kullanılır. Entropy[12] rastgeleliğin ve düzensizliğin hangi olasılıklarla ortaya çıkacağını gösterir. Olasılıkların eşit olduğu durumlar belirsizliğin yüksek olduğu anlamını taşır. Dolayısıyla verinin entropiyi en aza indirecek şekilde bölünmesi gerekir.

Entropy:

$$H = -\sum_{x}^{m} P(x)\log\mathrm{P}(x) \qquad \textbf{(3.1)}$$

Entropi denkleminde (3.1), p (x) belirli bir sınıfa ait grubun yüzdesini ve H ise entropiyi belirtmektedir.

Karar ağacımızın entropi değerini en aza indirgeyen bölünmeler yapmasını isteriz. En iyi bölünmeyi belirlemek içinde bilgi kazancını kullanırız. Bilgi kazancı (3.2) aşağıdaki eşitlik ile hesaplanır:

Bilgi kazancı:

$$\mathrm{Gain(S,D)} = H(S) - \sum_{k=0}^{n} \frac{|\mathrm{V}|}{|S|}\mathrm{H(V)} \qquad \textbf{(3.2)}$$

Burada, S orijinal veri kümesidir ve D ise kümenin bölünmüş bir parçasıdır. Her V, S'nin bir alt kümesidir. V'nin tümü ayrıktır ve S'yi oluşturmaktadır. Bu durumda bilgi kazancı, bölünmeden önceki orjinal veri setinin entropisi ile her bir özniteliğin entropi değeri arasındaki fark olarak tanımlanmaktadır. Entropi düzensizlik düzeyinin göstergesi iken, enformasyon oluşmuş ve oluşmakta olan düzenin derecesini ifade eder.

[12] Entropi (Claude Shannon, 1948.)

Karar ağaçlarında karar düğümlerini oluşturacak en iyi niteliğin seçilmesinin bir diğer sebebi kök düğümde ağacın dengeli bir şekilde dallanmasını ve sınıflandırma algoritmasının verimli olmasını sağlamaktır. [20]

Her bir yeni alt veri kümesi için en ayırt edici özellik belirleme işlemi özyinelemeli (recursive) olarak tekrarlanır. Özyinelemeli parçalama olarak adlandırılan bu süreç tekrarlama işleminin tahmin üzerinde etkisi kalmayana kadar sürer. Böylelikle her düğüm için veri seti örnekleri aynı gruba ait olmuş demektir. Artık örnekleri bölecek başka bir özellik kalmamıştır. Karar ağacının yaprak olarak ifade edilebilecek olan son değerleri ise elde edilen sonuçlardır.

Bu çalışmada ağaçlar, ara düğümlerde veri setinden girdi olarak alınmış metin parçacığının kelimeleri, yaprak düğümünde ise girdi olarak verilen verinin sınıf bilgisi ile öğrenme işlevini yerine getirmiştir. Parametrelerin optimizasyonu ise ızgara araması (grid search) algoritması ile gerçekleştirilmiştir.

3.1.4 Rassal orman sınıflandırma modeli

Rassal orman algoritması Bagging (Breiman, 1996) yöntemi ve Random Subspace (Tin Kam Ho, 1998) yöntemlerinin birleşiminden oluşmaktadır. Algoritmanın düşük sapma miktarlarıyla iyi tahminler üretmesinin sebebi seçilen ağaçlar arasındaki düşük korelasyondur. Bu etkiyi sağlayan en önemli faktör ağaçların birbirlerini tekil hatalardan korumalarıdır. Bazı ağaçlar hatalı olsa dahi, diğer birçok ağaç doğru olacaktır, böylece bir grup olarak ağaçlar doğru yönde hareket edecektir. Bu da birbirinden olduğunca farklı ağaçların seçimiyle mümkün olmaktadır. Rassal orman, adından da anlaşılacağı gibi, bir arada çalışan çok sayıda tekil karar ağacından oluşur. Her bir ağaç bir sınıf tahmininde bulunmakta ve en fazla oyu alan sınıf, modelin öngörüsü haline gelmektedir.

Rassal orman (Random Forest) algoritması denetimli bir sınıflandırma algoritmasıdır. Bu algoritma hem sınıflandırma hem de regresyon problemlerinde uygulanabilir olan, karar ağaçları temelli makine öğrenmesi modellerinden biridir. Rassal ormanı karar ağaçlarından ayıran en belirgin yön, karar ağaçlarının en büyük problemi olan aşırı öğrenme-veriyi ezberleme (overfitting) sorununun bu modelde çözülmüş olmasıdır. Rassal orman modelinde hem veri setinden hem de öznitelik setinden rassal olarak 10'larca farklı alt setler alınır ve bunlar eğitilir. Böylece birçok karar ağacı elde edilir ve her bir karar ağacı kendi tahminlerini yapar. İşte tam bu noktada karar ağaçlarının

en büyük problemi olan aşırı öğrenme sorunu çözülür. Bunun nedeni rassal orman modelinin farklı veri setleri üzerinde eğitim gerçekleştirmesi nedeniyle varyansı düşürmesidir. Nihayetinde regresyon problemi için karar ağaçlarının tahminlerinin ortalaması ve sınıflandırma problemi için tahminler arasından yüksek oy alan tahmin seçilir. [21]

Daha derine inildiğinde, rassal orman algoritmasında ağaçlar oluşturulurken Bootstrap örneklemeleri ve her düğüm ayrımında rastgele seçilen m adet tahmin edici kullanılır. Bootstrap metodunun temel mantığı mevcut veri setinden çok büyük veri setleri yapmak üzere yeniden örneklemeler almasıdır. Geliştirilen ağaçlar budanmaz (Archer, 2008; Beriman, 2001). Budamanın olmaması Rassal orman'ı diğer karar ağacı yöntemlerinden daha avantajlı hale getirmektedir. Eğer rassal orman sınıflandırma için kullanılacaksa ağaçlar, her yaprak düğümü sadece belirli bir sınıfın üyelerini barındıracak şekilde oluşturulurlar. Rassal orman sınıflandırıcısı ile bir ağaç üretmek için kullanıcı tarafından tanımlanan 2 parametre gereklidir. Bu parametreler, en iyi bölünmeyi belirlemek için her bir düğümde kullanılan değişkenlerin sayısı (m) ve geliştirilecek ağaçların sayısı (N)'dir (Pal, 2005). Bu m adet değişken her düğümde tüm değişkenler arasından rasgele seçilir ve aralarından en iyi olan dal belirlenir. Değişken sayısı m genelde toplam değişken sayısının kare köküne eşit alındığında en optimum sonuca ulaşılır. [21]

Oluşan sınıfların homojenlik kriteri GINI indeksidir. GINI indeksinin düşük olması o sınıfın homojen olmasına işaret eder. Bir alt düğüm bir üst düğümden daha düşük GINI indeksine sahipse o dal başarılı olarak yorumlanır. Tüm veri setinin GINI indeksi, veri setinde seçilen her bir veri için verinin kendisinden küçük ve kendisinden büyük eleman sayılarına bölümünün karelerinin toplamının 1'den çıkartılası ile hesaplanır. İndeks hesaplandıktan sonra test veri setlerinin sınıfları bu hesaplama baz alınarak belirlenir. Elde edilen sonuçlara ışığında en faydalı sınıflandırma yapılmış olur.[21]

Rassal ormandaki parametreler ya modelin tahmin gücünü arttırmak ya da modeli daha hızlı hale getirmek için kullanılır. Genel olarak, daha yüksek sayıda ağaç performansı artırır ve tahminleri daha kararlı hale getirir, fakat aynı zamanda hesaplamayı da yavaşlatır.

Önemli bir hiperparametre "max_features", yani rassal ormanın tek bir ağaçta denemesine izin verilen maksimum özellik sayısıdır. Hız açısından önemli olan diğer

hiperparametre, “min_sample_leaf” dir. Bu, adından da anlaşılacağı gibi, bir alt düğümü ayırmak için gereken minimum yaprak sayısını belirler. [21]

Ağaç yapısı Karar Ağaçları sınıflandırıcısındaki kullanıma özdeş şekilde, ara düğümlerde veri setinden girdi olarak alınmış metin parçacığının kelimeleri, yaprak düğümünde ise girdi olarak verilen verinin sınıf bilgisi ile öğrenme işlevini yerine getirmiştir. Parametrelerin optimizasyonu ise ızgara araması (grid search) algoritması ile gerçekleştirilmiştir.

3.1.5 Naive Bayes sınıflandırma modeli

Naive Bayes yöntemi bir sonucun ortaya çıkma ihtimalinin en yüksek olduğu koşula göre sonuç üreten bir sınıflandırma algoritmasıdır. Devamlı değişen dinamik veriler için test ve eğitim süreci tekrarlanacağından dolayı bu tür veri setleri için uygun bir algoritma değildir.

$$\boldsymbol{P(h|v) = \frac{P(v|h) * P(h)}{P(v)}} \qquad \textbf{(3.3)}$$

Algoritma tıpkı Bayes Teoremi’ndeki (3.3) aynı hesapla, eğitim fazı sayesinde önceden bilinen, yani önceden sınıflandırılmış bir veriden yola çıkarak yeni bir hipotezin olma olasılığını hesaplayabilmemiz için bize yol gösterir. P(h | v) ifadesi, v verisi bilindiğinde hipotez h‘nin olasılığıdır. Buna sonsal (posterior) olasılık denir. P(d | h) ifadesi, h hipotezinin doğru olduğu varsayıldığında v verisinin olasılığıdır. P(h), verinin ne olduğuna bakılmaksızın hipotezin olasılığıdır. P(v) ise hipotezden bağımsız olarak verinin olasılığıdır. Birçok hipotez için sonsal olasılık hesaplandıktan sonra bunlar arasından en yüksek olanı seçilir. Diğer bir deyişle v verisine P(h | v) olasılığı maksimum olan sınıfı atamaktır. Burada önemli noktalardan biri yöntemin P(v1, v2, v3 | h) şeklinde her bir özellik için olasılık hesaplamak yerine, özelliklerin koşullu bağımsız oldukları varsayımı ile hesap P(d1 | h)* P(d2 | h) şeklinde yapılmaktadır. Naive ön adı sınıflandırmayı etkileyen değişkenlerin birbirinden bağımsız olduğu varsayımını temsil eder.

Naive Bayes modelinde her bir sınıfın olasılıkları ve verilen her farklı girdi değerlerinin olasılıkları hesaplanır. Bu nedenle model, optimizasyon için herhangi bir parametreye ihtiyaç duymaz ve eğitim verilerinde öğrenmede oldukça hızlıdır.

Naive Bayes sınıflandırma algoritması GuassianNB, Multi-variate Bernoulli Naive Bayes ve Multinomial Naive Bayes olmak üzere üç farklı yaklaşım içermektedir.

3.1.5.1 GuassianNB sınıflandırıcı

Özniteliklerin normal bir Gauss dağılımına özdeş karakteristikte ve sürekli olduğu sınıflandırma modelidir.

3.1.5.2 Multi-Variate Bernoulli Naive Bayes sınıflandırıcı

Veri setindeki tüm özellik vektörlerini 1 ve 0 değerlerine eşlemektedir.

3.1.5.3 Multinomial Naive Bayes sınıflandırıcı

Bir terimin belgede bulunma sıklığı baz alınarak terim frekansına göre sınıflandırma yapılan modeldir.

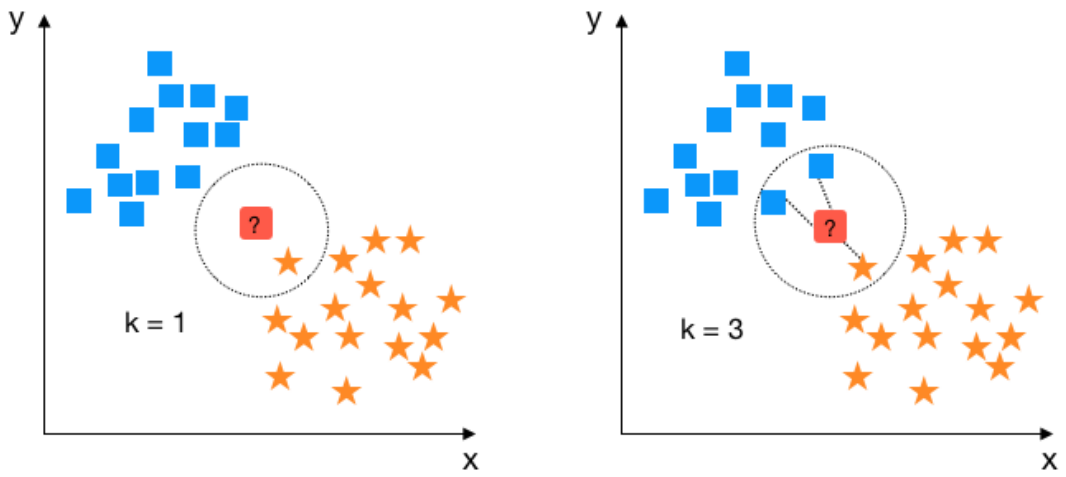
3.1.6 kNN sınıflandırma modeli

K-en yakın komşuluk algoritması (T. M. Cover, P. E. Hart, 1967) hem sınıflandırma hem de regresyon problemlerinin çözümünde kullanılan denetimli bir öğrenme algoritmasıdır. KNN algoritmasında sınıfı tahmin edilmek istenen veri ile mevcut veriler arasındaki uzaklıklar hesaplanır ve k sayıda yakın komşuluklarına göre sınıflandırma yapılır. Böylece yeni veriye yakın olan komşular uzak olanlara göre daha fazla katkıda bulunmuş olurlar. Uzaklıklar hesaplanırken öklid uzaklığı, manhattan uzaklığı, euclidean uzaklığı ve minkowski uzaklığı mesafe hesaplama yöntemleri kullanılmaktadır.

kNN algoritmasında komşuluk sayısı k, model oluşturulurken belirtilmesi gereken bir hiper parametredir. Her veri setinin kendine özgü gereksinimleri olacağı göz önünde bulundurulduğunda bir veri seti için optimum olan bir k sayısının her tür veri setine uymayacağı aşikardır. Az sayıda komşuluk sayısı seçilmesi durumunda gürültü sınıflandırma başarımı üzerinde yüksek bir etkiye sahip olacaktır. Çok sayıda komşu seçilmesi durumunda ise hesaplama maliyeti artmış olacaktır.

KNN algoritması gürültülü eğitim verilerine karşı dayanaklı olmasından dolayı popülerdir. Algoritma model oluşturmak için herhangi bir eğitim veri setine ihtiyaç duymaz, bu durum tembel algoritma olarak da ifade edilir. Test aşamasında tüm eğitim verisi kullanılır. Bu eğitimin hızlı olması anlamına gelirken zaman ve hafıza bakımından daha maliyetli bir test süreci anlamına gelir. Çünkü algoritma ile veriler

arasındaki uzaklık bilgilerinin tümünü saklandığından, büyük veriler üzerinde çalışılırken çok fazla bellek alanına ihtiyaç duyulur.



Şekil 3.3 : Yakın Komşuluk modeli k = 1 ve k = 3 komşulukları

3.1.7 Kolektif öğrenme sınıflandırma modeli (Ensemble Learning)

Kolektif öğrenme (Ensemble Learning) ya da diğer adıyla sınıflandırıcı toplulukları, sınıflandırma başarımını artırmak için birden çok sınıflandırma algoritmasını bir arada kullanma mantığını temel alan bir yapay öğrenme yöntemidir. Literatürde başta en yaygın kullanılan Destek Vektör Makinası, Karar Ağaçları, Rassal Orman, Yapay Sinir Ağları olmak üzere birçok sınıflandırma algoritması önerilmiştir. Bu sınıflandırıcılar tekil öğreniciler olarak da adlandırılır. Bu tekil öğrencilerin farklı veri setleri ve farklı sınıflandırma ihtiyaçları karşısında farklı başarımlar göstermektedir. Veri seti üzerinde uygulanan ön işlem adımları ve sınıflandırıcının uygun parametrelerle optimizasyonu sınıflandırma doğruluğunu seçilen tekil öğrenici için ancak belli bir noktaya kadar iyileştirebilmektedir. Kolektif öğrenme metodu, kullanılan eğitim verisinin yapısına uygun ve çeşitli denemeler yaparak bir model oluşturur ve tahminler verilen test verisi üzerinde belirlenen bu model ile gerçekleştirilir.

Bu sınıflandırma yönteminde temel amaç standart sapma ve varyansı azaltmaktır. Başlangıçta veri setinden yeni alt kümeler çıkartılır. Veri alt kümeleri farklı sınıflandırıcı algoritmalarla eğitilir ve sonuçlar bir araya getirilir. Burada başarılı sonuç almak için tekil öğrenicilerden gelecek sonuçlar da yüksek başarıma sahip olmalıdır. Bu kolektif sonucu daha da yukarı çekecektir. Kolektif öğrenme modelinin

başlıca çeşitleri Bagging, Boosting, Random Subspaces, Dagging ve Decorate metodlarıdır. [22]

3.1.7.1 Bagging sınıflandırıcı

Bu yöntemde, temel öğrenicilerin (base learner) her biri, eğitim setinin rastgele seçilen farklı alt kümeleriyle eğitilir. Bagging, "bootstrap aggregation" ifadesinden türetilmiştir. Burada temel fikir yeniden örneklem yöntemi ile tekrar tekrar örnekler çekerek yeni ağaçlar oluşturmaktır. Bu yeni ağaçlardan oluşan ağaç topluluğu modelleme için kullanılır ve her birinin tahminine başvurulur. Tahminlere göre bir ortalama belirlenir ve sonucunda tek bir tahmin değeri ortaya çıkar.

3.1.7.2 Boosting sınıflandırıcı

Boosting sınıflandırıcı da temel fikir birçok zayıf tekil öğreniciyi ardışık olarak bir araya getirip güçlü bir öğrenici meydana getirmektir. Gradient Boosting ve AdaBoost metodları Boosting sınıflandırıcının en bilinen ve sık kullanılan metodlarıdır. AdaBoost metodunun temel dinamiği kendinden önce gelen bir tahmin edici modelin öğrenmede geri kaldığı eğitim verilerine daha fazla önem vermesidir. Gradient Boosting metodunda ise zayıf tahmin modellerinin bir araya gelmesiyle tipik olarak karar ağaçlarının oluşturduğu bir model oluşturur. Tahmin ediciler tekrar eğitildikten sonra başarımlarına göre sıralanırlar ve bu noktadan sonra tahminler Bagging metodu ile yapılır.

3.1.7.3 Extremly randomized trees

Aşırı Rassal Ağaçlar sınıflandırıcı (Geurts et al., 2006) temelde Rassal Orman sınıflandırma algoritmasına çok benzemekle birlikte karar ağaçlarının inşası bakımından farklılık gösterir. Birincisi sınıflandırıcıyı eğitmek için birden fazla ağaç seçilir ve her ağaç eğitim verilerinin tümü kullanılarak eğitilir. İkincisi ise modelin bir düğümü parçalara ayırırken en önemli özelliği aramak yerine, rastgele bir özellik alt kümesi seçmesidir. Bu, genellikle daha iyi bir modelle sonuçlanan geniş bir çeşitlilikle sonuçlanır ve bu artan rastgelelilik varyansın Rassal Orman'a kıyasla daha da düşmesini sağlar. [23]

3.1.7.4 Voting sınıflandırıcı

Voting sınıflandırıcı farklı sınıflandırma modellerinden gelen tahminleri bir araya

getiren ve bu değerler arasından seçim yaparak nihai tahmini elde edildiği bir yaklaşımdır. Genel bir ifadeyle, her algoritmanın farklı yönlerini bir araya getiren bir yaklaşımdır ve bu nedenle gerçek bir sınıflandırma algoritması değildir. Nihai çıktı tahminler arasından oy çokluğu prensibi ile seçilir.

3.1.8 OneVSRest sınıflandırıcı

Bu yaklaşım çoklu sınıflı sınıflandırma için yaygın kullanılan bir stratejidir. Temel fikir, çoklu sınıflı bir sınıflandırmada problemin ikili sınıflandırma olarak ele alınıp sınıflardan birinin diğerlerinden ayrılması ve kalan tüm sınıfların tek bir sınıf etiketi altında toplanmasıdır. Tüm ikili sınıflandırıcıların çalıştırılması ve en güvenilir tahminin seçilmesiyle nihai sonuca ulaşılmış olur.

3.2 Sınıflandırma Modelinin Ölçüm Kriteri

Bu çalışmada inşa edilen sınıflandırma modellerinin başarımlarının test edilmesi için hata metrisindeki dört temel değer ile hesaplanan ölçüm değerleri kullanılmıştır. Hata matrisi doğru pozitif, yanlış pozitif, doğru negatif ve yanlış negatif olmak üzere dört değerden meydana gelir.

Şekil 3.4 ‘de hata matrisini meydana getiren kavramlar çalışmada temsil ettikleri anlamlar ile tanımlanmıştır. Burada kullanılan “bir web sitesinin sınıfı” ifadesi o web sitesinin konusunu ve web ortamında en muhtemel hizmet alanını temsil etmektedir.

Doğru Pozitif (True positive, **TP**)	Eğitim kategorisindeki bir web sayfasına Eğitim kategorisinde olduğunu tahmin etmek
Yanlış Pozitif (False positive, **FP**)	Eğitim kategorisinde olmayan bir web sayfasına Eğitim kategorisinde olduğunu iddia etmek
Yanlış Negatif (False negative, **FN**)	Eğitim kategorisindeki bir web sayfasına Eğitim kategorisinde olmadığını iddia etmek
Doğru Negatif (True negative, **TN**)	Eğitim kategorisinde olmayan bir web sayfasına eğitim kategorisinde olmadığını tahmin etmek

Şekil 3.4 : Hata matrisi kavramlarının web sınıflandırması temsilleri ile açıklanması.

Bu değerler İngilizce karşılıklarının baş harfleri olan TP, FP, TN, FN kısaltmalarıyla temsil edilmektedir.

		Gerçek Değerler	
		Pozitif	Negatif
Tahmin Değerleri	Pozitif	**TP**	**FP**
	Negatif	**FN**	**TN**

Şekil 3.5 : Hata matrisi.

Bu tanımlama iki sınıflı bir sınıflandırıcı için temel ifadedir. Çalışmamızda 17 kategori bulunduğundan dolayı Şekil 3.5'deki matris 17 satır ve sütundan oluşur.

Hata matrisini meydana getiren bu dört değer çeşitli kombinasyonlarla formüle edildiğinde başarı ölçüm kriterleri elde edilir. Bunlar duyarlılık(recall), kesinlik(precision) ve F1 skorudur.

Duyarlılık tüm pozitif sınıflardan ne kadar doğru tahmin edildiğini gösterir. (3.4)

$$Duyarlılık\ (Recall) = \frac{TP}{TP+FN} \quad \textbf{(3.4)}$$

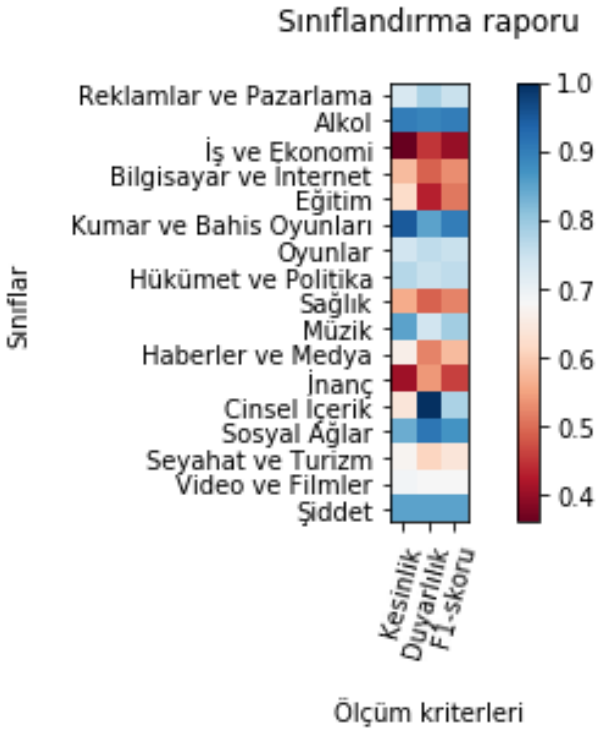
Kesinlik tüm sınıflardan ne kadar doğru tahmin edildiğini gösterir. (3.5)

$$Kesinlik\ (Precision) = \frac{TP}{TP+FP} \quad \textbf{(3.5)}$$

F-skor duyarlılık ve kesinlik değerleri aynı anda ölçmeyi sağlar ve bu iki değerin harmonik ortalamasıdır. (3.6)

$$F1 - skoru = \frac{2\ x\ Duyarlılık\ x\ Kesinlik}{Duyarlılık+Kesinlik} \quad \textbf{(3.6)}$$

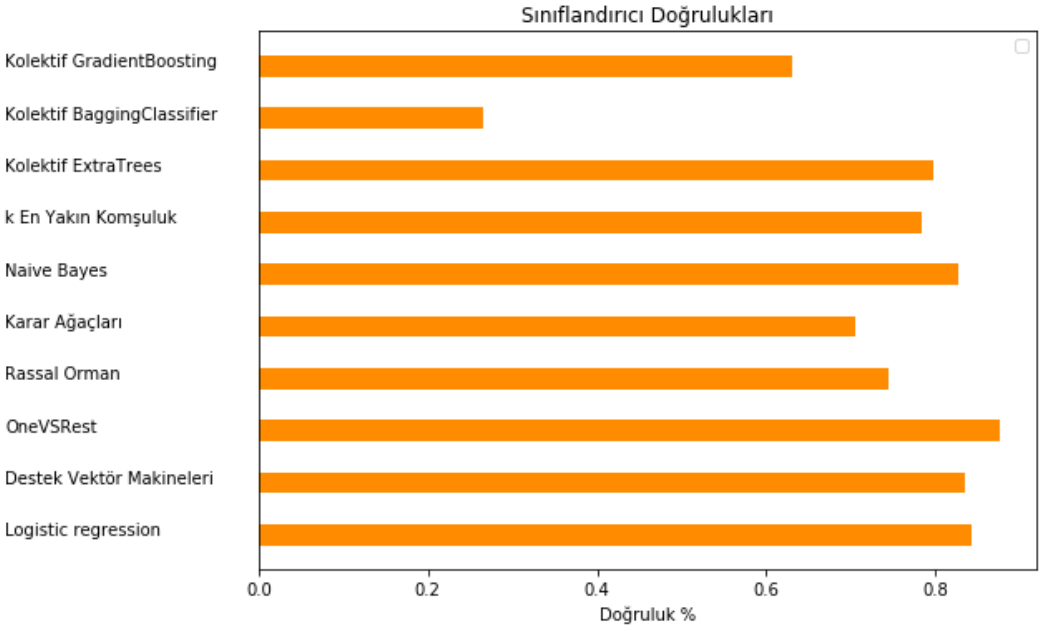
Şekil 3.6'de çalışmada oluşturulmuş bir sınıflandırma modeli için tahmin sonuçları ile elde edilen kriterler sınıf bazlı olarak görselleştirilerek sunulmuştur.



Şekil 3.6 : Ölçüm kriterleri türündeki sınıflandırma raporunun görselleştirilmiş formu.

4. DENEYSEL SONUÇLAR VE ANALİZ

Çalışmanın sonucunda web sitelerinin ham metinleri üzerinden elde edilen bir veri seti ile çoklu sınıflı sınıflandırma yapılmış ve mevcut sınıflandırma modelerinin başarımları gözlenmiştir. Sonuçlar göz önüne alındığında en yüksek sınıflandırma başarımını sırasıyla OnevsRest (%84), Lojistik Regresyon (%84) ve Gradient Boosting (%83) algoritmaları göstermiştir. Diğer algoritmalar birbirlerine oldukça yakın doğruluk oranlarına sahiptir.



Şekil 4.1 : Sınıflandırma algoritmaları başarı karşılaştırması.

Şekil 4.1'de sınıflandırma algoritmalarının başarı karşılaştırılmaları görsel olarak sunulmuştur. Sayısal veriler ve detaylı karşılaştırmalar Çizelge 4.2'de yer almaktadır.

Önişleme deneysel araştırma adımlarının başarıma etkileri yöntem bazında sunulmuştur. Tüm bu sonuçlar Çizelge 4.2'da yer almaktadır. Önişleme adımlarında deneysel olarak önerilen 3 yöntem ile elde veri setlerinin sınıflandırma başarımları üzerindeki etkileri analiz edildiğinde;

- Kaynak dilinin korunduğu 1. Yöntem ve çeviri işlemenin doğal dil işleme süreçlerinden önce yapıldığı 2. Yöntem, dil bazlı doğal dil işleme süreçlerinin çeviri işleminden önce yapıldığı 3. Yönteme göre daha yüksek başarım elde etmiştir
- 1.Yöntemi 3. Yöntemden üstün kılan en belirgin faktörün öznitelik çeşitliliği olduğu öngörülmektedir. 2. Yöntem'de ise 3. Yöntem'e benzer şekilde çeviri işlemi uygulanmasına karşın daha iyi bir sınıflandırma sonucu elde edilmiştir. Bunun öngörülen başlıca sebebi gövdeleme ve anlamsal kök elde etme işleminin çeviri başarımını düşürerek cümleleri ve kelimeleri kullanıldıkları anlamlardan uzaklaştırmasıdır. 2. Yöntem'de çeviri fonksiyonuna girdi olarak verilen veri noktalama işaretlerinin korunduğu anlamlı cümlelerdir.
- Ancak 3. Yöntem'de noktalama işaretleri korunmuş olsa dahi kelimeler kök çekimlerine indirgendiğinden dolayı çeviri fonksiyonun girdi verisindeki kelimeler cümle içi anlamını kaybetmiş, kelime çantası formatına benzer şekilde değerlendirilmiştir. Tüm bu sonuçlar analiz edilerek, sonuçlar Şekil 3'te görselleştirilmiştir.

Çalışma kapsamında, belirlenen 7 dil üzerinde her bir dile ait metinlerin, 3 önişleme yöntemi sırasıyla uygulanarak, ayrı ayrı bir araya getirilmesiyle oluşturulan veri setleri ile sınıflandırma yapılarak veri önişleme süreçlerinin dil bazlı analizi yapılmıştır. En yüksek başarımı sağlayan Logistik Regresyon, OneVsRest ve Naive Bayes sınıflandırıcıları bu veri setleri ile eğitilmiş ve böylelikle dil özelinde çeviri başarımları analiz edilmiştir. Sonuçlar Çizelge 4.1'da sunulmuştur.

Sınıflandırmada iki farklı vektörleştirme yöntemi kullanılmıştır. Her bir vektörleştirme yöntemi için tüm sınıflandırma modelleri üzerinde sonuçlar incelenmiştir. Karar ağaçları dışında tüm sınıflandırma modellerinde kelime torbası vektörleştirme, TFxIDF vektörleştirmeden daha yüksek başarı sağlamıştır.

Sınıflandırma modellerinin veri seti üzerinde kazanabileceği en yüksek öğrenme yeteneğini gözlemlemek için çapraz doğrulama yöntemi kullanılır. Sınıflandırma modelleri için optimum parametreler çapraz doğrulamam ile hesaplanmıştır.

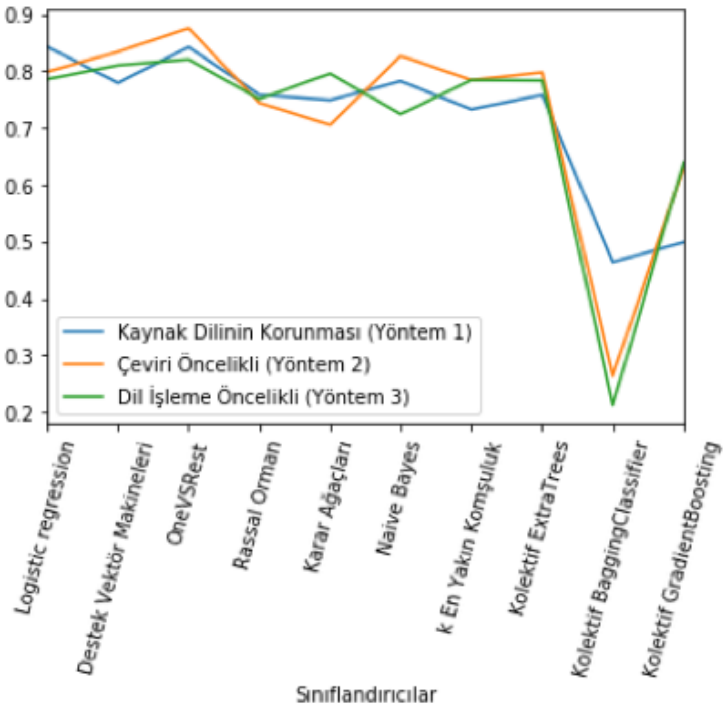
Veri seti boyutlarının sınıflandırma başarısı üzerindeki etkilerinin gözlemlenmesi için en başarılı sınıflandırma sonucunu veren sınıflandırma modelleri farklı boyutlardaki veri setleri üzerinde eğitilmiş ve sonuçlar karşılaştırılmıştır. Yöntem ve sınıflandırma

Çizelge 4.1 : Dillere göre çeviri başarılarının karşılaştırılması (Yöntem 1) Kaynak dilinin korunması, (Yöntem 2) Çeviri öncelikli, (Yöntem 3) Dil işleme öncelikli.

Sınıflandırıcı	Önişleme Yöntemi	İngilizce	Türkçe	Fransızca	İspanyolca	Almanca	Rusça	Portekizce
Lojistik Regresyon	Yöntem 1	0.7588	0.7522	0.8533	0.8034	0.7132	0.6810	0.8115
	Yöntem 2	0.7478	0.7888	0.8892	0.8327	0.7598	0.6520	0.8074
	Yöntem 3	0.7033	0.7556	0.8533	0.8037	0.7265	0.6239	0.8012
NBayes	Yöntem 1	0.7101	0.7334	0.8703	0.8425	0.6477	0.6505	0.8340
	Yöntem 2	0.7486	0.7142	0.8636	0.8266	0.6324	0.6993	0.8636
	Yöntem 3	0.6985	0.7730	0.8934	0.8626	0.7078	0.6439	0.8179
OneVSRest	Yöntem 1	0.7841	0.7731	0.8952	0.8368	0.7957	0.6354	0.8412
	Yöntem 2	0.7836	0.7455	0.8646	0.8785	0.7567	0.6828	0.7923
	Yöntem 3	0.8030	0.7202	0.8409	0.8565	0.6645	0.6256	0.8032

algoritması bazlı sınıflandırma sonuçlarının görsel formu Şekil 4.2'de sunulmuştur. Sonuçlara Çizelge 4.2'da detaylı şekilde yer verilmiştir. Bu işlemler sırasında her veri seti test ve eğitim seti olarak aynı oranda ayrıştırılmıştır.

Metin bazlı sınıflandırmada özellik çıkarımı kelimeler ile yapıldığı için eğitimde kullanan özellik matrisi çok büyük boyutlara ulaşacaktır. Bu problemin veri seti karakteristiği bozulmadan ve hiçbir özellik kaybedilmeden çözülebilmesi için kullanılan yaygın yöntemlerden birisi de normalleştirmedir. Normalleştirme uygulamasının sınıflandırma başarımı üzerindeki etkileri incelenmiştir.



Şekil 4.2 : Boyut indirgeme metotlarının sınıflandırma başarımı üzerindeki etkileri.

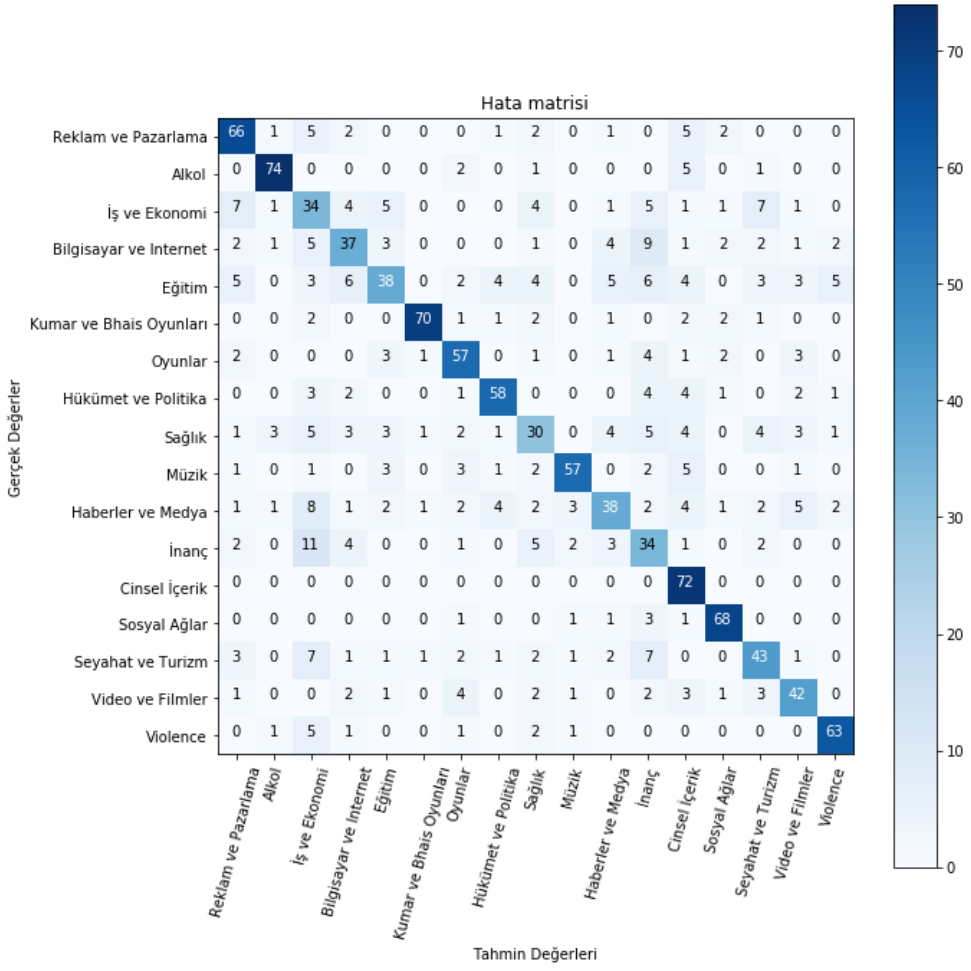
Lemmatization ve Stemmer doğal dil işleme çalışmalarında sıklıkla kullanılan iki yöntemdir ve özelik matrisinin boyutunu indirgemek için kullanılırlar. Bu iki boyut indirgeme yöntemi tüm sınıflandırma modellerinde incelenmiştir. Detaylı sayısal sonuçlar Çizelge 4.2'da yer almaktadır.

Çoklu sınıflı sınıflandırmada başarımı etkileyen en önemli faktörlerden bir diğeri bir metnin birden fazla sınıfa benzer ağırlıklarla yakınlık göstermesidir. Bu durum doğrusal sınıflandırma da iki sınıfa ait verilerin hiper düzleme en yakın olduğu noktalarda, diğer modellerde ise hiper düzlemin, verinin farklı sınıflara ait kısımlarının kurduğu aşırı yakınlıktan dolayı tam ayrıştırma yapamadığı durumlarda ortaya çıkar ve karışıklığa sebep olmasıdır. Bu ve bundan sonraki çalışmalarda sınıf etiketlerinin sayı ve çeşidinin belirlenmesinde önemli bir referans olabilmesi için çalışmadaki en yüksek başarımlı eğitim modelinin hata matrisi de Şekil 4.3'da sunulmuştur. Bu hata matrisi sayesinde birbirine en çok yakınlık gösteren sınıflar belirlenmiştir. Çalışma 26 sınıf ile başlamış, hata matrisinin okunup değerlendirilmesi ve önceliğin zararlı web sitelerinin sınıflandırılması oluşundan dolayı sınıf sayısı 17 olarak belirlenmiştir.

Bu çalışma, metin temizliği, sınıflandırma modellerinin seçimi ve onlara ait optimum parametrelerin belirlenmesi, boyut indirgeme modellerinin tercihinde ve sonuçları değerlendirilmesi gibi konularda bundan sonra yapılacak metin bazlı sınıflandırma çalışmalarda yararlı bir referans olacaktır.

Multinomiyal lojistik regresyon algoritmasında kullanılan Maksimum Olabilirlik Yaklaşımı veri seti ile logaritmik eğri arasında uyum sağlamak için modeli dağılım ortalama değerinin olduğu bölgeye kaydırmasının en yüksek başarıyı elde etmede önemli etken olduğu öngörülmektedir.

OneVSRest algoritması temelinde ikili sınıflandırma mantığına dayalı bir çoklu sınıflandırma metodolojisi sunar. Web sitesi metni üzerinde ikili sınıflandırmanın analiz edildiği "Metin Madenciliği ve Makine Öğrenmesi ile İnternet Sayfalarının Sınıflandırılması" [1] konulu çalışmaya dayanarak ikili sınıflandırma başarımının %91 ve üzeri başarıma sahip olduğu sonucu bu neticeyi desteklemektedir.



Şekil 4.3 : Hata Matrisi. Bu hata matrisi Lojistik Regresyon sınıflandırma modeli ile eğitilen yapının karışıklık ölçüsünü ifade eder.

Çizelge 4.2 : Sınıflandırma modelleri doğruluk karşılaştırma tablosu. Her sınıfa ait 250'şer olmak üzere toplam 4250 web sayfası metninden oluşan küçük veri seti ve toplam 35000 web sitesinden oluşan büyük veri seti üzerinde gerçekleştirilen eğitim ve test faaliyetlerine dair sınıflandırma başarımlarının karşılaştırma tablosu.

Sınıflandırıcılar	Kaynak Dilinin Korunması (Yöntem 1)				Çeviri Öncelikli (Yöntem 2)				Dil İşleme Öncelikli (Yöntem 3)			
	Stemming		Lemmatizasyon		Stemming		Lemmatizasyon		Stemming		Lemmatizasyon	
	Küçük veri	Büyük veri	Küçük veri	Büyük veri	Küçük veri	Büyük veri	Küçük veri	Büyük veri	Küçük veri	Büyük veri	Küçük veri	Büyük veri
Logistic regression	0.7376	0.8344	0.7276	0.8432	0.7434	0.7980	0.7509	0.7987	0.7557	0.7589	0.7524	0.7856
Destek Vektör Makineleri	0.6889	0.7547	0.6823	0.7791	0.7367	0.7912	0.7002	0.8341	0.6998	0.8123	0.7112	0.8102
OneVSRest	0.7841	0.7658	0.7748	0.8424	0.7666	0.8569	0.8032	0.8756	0.7642	0.8203	0.8071	0.8205
Rassal Orman	0.7315	0.7233	0.7423	0.7588	0.701	0.7105	0.7499	0.7445	0.7173	0.7389	0.7814	0.7511
Karar Ağaçları	0.6838	0.7439	0.6923	0.7491	0.7045	0.7566	0.6771	0.7053	0.6886	0.7673	0.6919	0.7956
Naive Bayes	0.7115	0.7113	0.7122	0.7823	0.7551	0.7734	0.7368	0.8268	0.7433	0.8032	0.7133	0.7245
k En Yakın Komşuluk	0.6504	0.7399	0.6501	0.7329	0.6743	0.7337	0.6523	0.7842	0.6438	0.7774	0.6442	0.7843
Kolektif ExtraTrees	0.7123	0.7565	0.7054	0.7582	0.6879	0.7455	0.7509	0.7977	0.7075	0.7109	0.6907	0.7834
Kolektif BaggingClassifier	0.7267	0.7932	0.4198	0.4643	0.4556	0.5233	0.7002	0.2645	0.4847	0.2896	0.6851	0.2125
Kolektif Grad. Boosting	0.4859	0.5405	0.4112	0.4990	0.7490	0.8312	0.8032	0.6308	0.5791	0.6302	0.5124	0.6389

5. GELECEK ÇALIŞMALAR

İleriki çalışmalarda doğal dil işleme yöntemleri kullanılarak web sitesi içeriklerinin anlam bakımında ve dil özelliklerine göre sınıflandırılması hedeflenmektedir. Bu kapsamda mevcut veri seti üzerinde noktalama işaretleri de korunarak dilbilgisinin anlam odaklı sınıflandırma üzerindeki etkileri de cümle bazlı olarak takip edilebilecektir.

Günümüzde zararlı web siteleri metinlerden bağımsız yalnızca görüntü içerikli olabilmektedir. Bu doğrultu da diğer çalışma alanı da web sitelerindeki görsellerin depolanıp bu görseller üzerinde görüntü işleme ile sınıflandırma yapılmasıdır.

KAYNAKLAR

[1] **Şahin**, İ. (2019). Metin Madenciliği ve Makine Öğrenmesi İle İnternet Sayfalarının Sınıflandırılması, Hacettepe Üniversitesi Endüstri Mühendisliği Anabilim Dalı Yüksek Lisans Tezi.

[2] **Virtanen, E. E.** (2019). Classification of Web Elements Using Machine Learning, Aalto Üniversitesi İletişim ve Ağ İletişimi Bölümü.

[3] **Tüfekci, P., Uzun, E. and Sevinç**, **B.** (2012). Text classification of web based news articles by using Turkish grammatical features, 20th Signal Processing and Communications Applications Conference (SIU), Mugla, 2012, pp. 1-4.

[4] **Gürcan, F.** (2009). Web İçerik Madenciliği ve Konu Sınıflandırılması, Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsü Yüksek Lisans Tezi.

[5] **Güven E. N., Onur, H., Sağıroğlu, Ş.** (2007). Yapay Sinir Ağları ile Web İçeriklerini Sınıflandırma, Değişen Dünyada Bilgi Yönetimi Sempozyumu, 24-26 Ekim 2007, Ankara.

[6] **Indra, M., Karuppasamy, S. and Rajaram, R.** (2009). Fast web page classification without accessing the web page using machine learning techniques, Journal of Information, Intelligence and Knowledge. 1.

[7] **Patil, A. S., Pawar, B.V**. (2012). Automated Classification of Web Sites using Naive Bayesian Algorithm, Proceedings ıf the International MultiConference of Engineers and Computer Scientitsts 2012 Vol 1, IMECS 2012, March 14-16, Hong Kong.

[8] **Lan, M., Tan, C. L., Jian, S. and Lu, Y.** (2019). Supervised and Traditional Term Weighting Methods for Automatic Text Categorization, IEEE Transactions on Pattern Analysis and Machine Intelligence 31(4):721-35.

[9] **Fayyad, U., Shapiro, G., Smyth**, **P.** (1996). From Data Mining to Knowledge Discovery in Databases, American Associatin for Artificial Intelligence, cilt 17, s. 37-54.

[10] **Debole, F., Sebastiani**, **F.** (2003). Supervised term weighting for automated text catego-rization, In Proceedings of SAC-03, 18th ACM Symposium on Applied Computing. pp. 784–788. ACM Press

[11] **Vinciarelli, A., Luettin**, **J.** (2001). A new normalization technique for cursive handwritten words, Pattern Recognition Letters, Volume 22, Issue 9.

[12] **Wei, J., Chang, C., Chou N. and Jan, G.** (2001)**.** ECG data compression using truncated singular value decomposition," in IEEE Transactions on Information Technology in Biomedicine, vol. 5, no. 4, pp. 290-299, Dec.

[13] **Yen, I. E. H., Xiangru, H., Pradeep, R. and Inderjit**, **D.** (2016). PD-Sparse: A Primal and Dual Sparse Approach to Extreme Multiclass and Multilabel Classification, Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, JMLR: W&CP volume 48.

[14] **Hutcheson, G. D**., (2011). Ordinary Least-Squares Regression. In L. Moutinho and G. D. Hutcheson, The SAGE Dictionary of Quantitative Management Research. Pages 224-228.

[15] **Aldrich**, **J.** (1997). Statistical Science, Vol. 12, No. 3, 162-176

[16] **Suykens, J. A., Vandewalle**, **J.** (1999). Least squares support vector machine classifiers. Neural processing letters, 9(3), 293-300.

[17] **Boser, B. E., Guyon, I. M. and Vapnik, V. N.** (1992). A training algorithm for optimal margin classifiers, Proceedings of the fifth annual workshop on Computational learning theory. ACM.

[18] **Hssina, B., Merbouha, A., Ezzıkourı, H. and Errıtalı, M.** (2014). A comparative study of decision tree ID3 and C4, 5. International Journal of Advanced Computer Science and Applications 4.2.

[19] **Özekeş, S., Çamurcu, A.** Y. (2002). Veri Madenciliğinde Sınıflama ve Kestirim Uygulaması, Marmara Üniversitesi Fen Bilimleri Dergisi, sayı 18, s. 1-17.

[20] **Ethem**, **A.** (2014), Introduction to Machine Learning, Chapter 9: Decisipn Trees, s. 185-207

[21] **Andy, L., Wiener**, **M**. (2002). Classification and regression by randomForest, *R news* 2.3: 18-22.

[22] **Bulut. F.** (2016). Sınıflandırıcı Topluluklarının Dengesiz Veri Kümeleri Üzerindeki Performans Analizleri, İzmir Kâtip Çelebi Üniversitesi, Bilgisayar Mühendisliği, İzmir.

[23] **Götz, M., Weber**, **C., Blöcher, J., Stieltjes, B., Meinzer, H., and Maier-Hein, K.** (2014). Extremely randomized trees based brain tumor segmentation.

[24] **Hackeling, G.** (2014). Mastering Machine Learning with scikit-learn, Chapter 3: Feature Extraction and Preprocessing, 51-69, Chapter 5: Nonlinear Classification and Regression with Decision Trees, 97-112, Chapter 9: From the Perceptron to Support Vector Machines, 172-182.

[25] **Mckinney. W**. (2012) Python for Data Analysis, Chapter 4 NumPy Basics: Arrays and Vectorized Computation, 80-104, Chapter 6 Data Loading, Storage, and File Formats, 155-166.

[26] **Çalış, A., Kayapınar, S. and Çetinyokuş, T.** (2014). Veri Madenciliğinde Karar Ağacı Algoritmaları İle Bilgisayar ve İnternet Güvenliği Üzerine Bir Uygulama, Endüstri Mühendisliği Dergisi Sayı: 3-4 Sayfa: (2-19).

[27] **Plisson, J., Nada L. and Mladenic**, **D.** (2004). A rule based approach to word lemmatization, Proceedings of IS-2004: 83-86.

[28] **Vijayarani, S., Ilamathi, J. and Nithya, Ms.** (2015). Preprocessing techniques for text mining-an overview, International Journal of Computer Science & Communication Networks 5.1: 7-16.

[29] **Url-1** <http://bilgisayarkavramlari.sadievrenseker.com/2013/02/08/naif-bayes-siniflandiricisi-naive-bayes/>, alındığı tarih: 15.09.2019.

[30] **Url-2** <http://cagriemreakin.com/veri-bilimi/deep-learning/ann.html>, alındığı tarih: 11.07.2019.

[31] **Url-3** <https://medium.com/data-science-tr/makine-öğrenmesi-dersleri-4a-lineer-regresyon-30fe61248e93>, alındığı tarih: 13.07.2019.

EKLER

EK A: Veri setinin kısmi görünümü

EK B.1: Web sayfalarının sorgulanmasını ve ham metinlerin diske kaydedilmesini sağlayan Python betiği.

EK B.2: Veri önişleme, boyut indirgeme ve temizlik işlemlerini kapsayan Python betiği. Bu işlemler tamamlandığında sınıflandırma ve eğitim işlemlerinde kullanılacak olan veri seti oluşmuş olur.

EK B.3: Oluşturulan veri setinden eğitim ve test veri setlerinin oluşturulduğu, vektörleştirme, normalizasyon, çapraz doğrulama, görselleştirme, sınıflandırma algoritmaları ile model inşası ve tahmin sonuç analizlerinin çıkarımı gibi işlemlerin yapıldığı Python betiği.

EK C.1: Çapraz doğrulama işlemleri sonucu elde edilen optimal parametler. Bu sonuçlar Python programlama dilinin sklearn kütüphanesinin GridSearchCV modülü ile gerçekleştirilmiştir.

EK C.2: Önemli sınıflandırıcı paremetreleri ve açıklamaları.

EK A.1

Sınıf İsimleri	Web sayfası	İçerik
Education	http.www.citesandinsights.info.txt	large large policy technology media all contents search back about contact and feedback pay what you wish does not have sponsorship support does have both direct out pocket money and indirect you find valuable you can help keep going the button below secure link that you can donate money credit card pay what you wish what you think worth suggest you feel specific issue worth and more for year worth you don think worth just can afford consider standing payment month good month better month would great current issue the front feedback domain checked once twice day issue atom feed for update update issue and infrequent special
Gambling	http.www.surferspoker.com.txt	surfers poker poker forum bonus codes rakeback wsop poker strategy poker glossary best online poker players biggest online poker wins wcoop pokerstars william hill carbon poker cake poker poker promotions poker sites pokersite comparisons poker quizzes starting hand odds brm chart poker news and articles wsop events what new party poker rakeback titan bonus surfers poker poker forum user name remember password register search today posts mark forums read welcome the surfers poker poker forum welcome surfers poker poker here poker forum viewing general poker discussion years venice going deep viewing
Sexual Content	http.www.xxx.com.txt	free sex tube pussy free and sex content search best hot sex forum all sex more full list biz jump category more full list more hot sex more free showing most popular total naked with son min horny husband let wife hard with min bet that could make friend come less than and won min the came off and got cum her pussy min ana man min min not only loving but also beautiful black side piece riley king min big booty back and like champ min khalifa cherry and was awkward min good looking blonde real estate agent sex apartment min anal for young skinny teen really big cock deep her and she make her master happyt threesome with lolly ink min petite wife min special such good whore min queen min big boob black teen girl jenna older white man tommy min petite brunette lily soapy shower sex min painful
Health	http.www.childliverdisease.org.txt	liver disease foundation home about news contact shop menu support liver information how help donate the marathon that everyone can complete run walk cycle share cover throughout may and help make difference learn more liver disease foundation fighting childhood liver disease providing information emotional support research funds and voice for all affected you are parent child with liver disease click here find out about the information and support available you you are aged with liver condition find out here how can help you you are health professional whether you are part hospital based team working the community find out how can help you you would like support our work discover the ways which you can make real difference with liver disease across the information childhood liver disease from help
Social Networking	http.www.meetandmate.com.txt	meetandmatecom online personals release your sexual inhibitions man woman searching for man woman alabama alaska arizona arkansas california colorado connecticut delaware district columbia florida georgia hawaii idaho illinois indiana iowa kansas kentucky louisiana maine maryland massachusetts michigan minnesota mississippi missouri montana nebraska nevada new hampshire new jersey new mexico new york north carolina north dakota ohio oklahoma oregon pennsylvania rhode island south carolina south dakota tennessee texas utah vermont virginia washington west virginia wisconsin wyoming men seeking women women seeking men fairbanks juneau birmingham montgomery jonesboro little rock flagstaff phoenix los angeles sacramento san francisco denver pueblo bridgeport hartford dover newark
Alcohol	http.www.beerbrew.com.txt	homebrew emporium online quality products expert advice skip navigation skip content homebrew emporium modern homebrew emporium south shore homebrew emporium west boylston homebrew emporium westchester homebrew emporium search for shop newsletter newsletter archive blog recipes resources classes resources classes rensselaer locations homebrew emporium west boylston homebrew emporium modern homebrew emporium south shore homebrew emporium westchester homebrew emporium search for navigation homeblog cart checkout class schedule homebrew emporium class schedule south shore homebrew emporium class schedule locations homebrew emporium rensselaer modern homebrew emporium cambridge south shore homebrew emporium south weymouth west boylston homebrew emporium west boylston westchester homebrew emporium new rochelle subscription authenticated subscription

Şekil A.1: Veri seti kısmi örnek

EK B.1

```
1.  import urllib.request
2.  import urllib
3.  import ssl
4.  import chardet
5.  import re
6.  from socket import timeout
7.  from urllib.error import HTTPError, URLError
8.  import logging
9.  ssl.match_hostname = lambda cert, hostname: True
10.
11. with open('urlData.txt') as f:
12.     lines = f.readlines()
13.     for line in lines:
14.         print(line)
15.         url = b''
16.         url = line.split(',', 1)[1]
17.         tmp = b''
18.         tmp = line.split(',', 1)[0]
19.         cat = b''
20.         cat = tmp.rstrip()
21.         httpurl = b''
22.         httpurl= "http://www."+url
23.         try:
24.             html = b''
25.             rawdata = b''
26.             charlan = b''
27.             charenc = b''
28.             charcon = b''
29.             user_agent = 'Mozilla/5.0 (Windows; U; Windows NT 5.1; en-
    US; rv:1.9.0.7) Gecko/2009021910 Firefox/3.0.7'
30.             headers={'User-Agent':user_agent,}
31.             try:
32.                 request=urllib.request.Request(httpurl,None,headers) #The a
    ssembled request
33.                 response = urllib.request.urlopen(request,timeout=10)
34.                 rawdata = response.read()
35.             except (HTTPError, URLError) as error:
36.                 logging.error('Data not retrieved because %s\nURL: %s', err
    or, url)
37.             except timeout:
38.                 logging.error('socket timed out - URL %s', url)
39.             else:
40.                 logging.info('Access successful.')
41.             print(rawdata)
42. """
43. Chardet kütüphanesi ile ham metnin dil ve kodlama türü bilgisinin temini
44. """
45.             result = chardet.detect(rawdata)
46.             print(result)
47.             charlan = result['language']
48.             charenc = result['encoding']
49.             charcon = result['confidence']
50. """
51. Chardet sonuçları %10 veya daha az hata oranına sahipse sonuçları dikkate a
    l.
52. """
53.             if (charcon > 0.90):
54.                 try:
```

```
                    html = rawdata.decode(charenc)
                except (HTTPError, URLError) as error:
                    logging.error('Data not retrieved because %s\nURL: %s',
    error, url)
                except timeout:
                    logging.error('socket timed out - URL %s', url)
                else:
                    logging.info('Access successful.')
            else:
                try:
                    html = rawdata.decode("utf8", "ignore")
                except (HTTPError, URLError) as error:
                    logging.error('Data not retrieved because %s\nURL: %s',
    error, url)
                except timeout:
                    logging.error('socket timed out - URL %s', url)
                else:
                    logging.info('Access successful.')
"""
Ham metnin web sitesi ismi ve kategori ismini içeren
dosya ismi ile metin dosyası olarak diske kaydedilmesi
"""
            print(html)
            outfilename =re.sub("[^a-zA-Z0-9.:]+", "", str(url))
            print(outfilename)
            mypath = r"/wisimo"
            filename= (mypath + str(cat) + "," + str(outfilename) + ".txt")

            print(filename)
            with open(filename, 'w') as f:
                print('', html, file=f)  # Python 3.x

        except IOError:
            pass
```

EK B.2

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Sat Dec 22 18:07:01 2019

@author: kenanaydin
"""
import re
import html2text
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
from nltk.stem import SnowballStemmer
from nltk import wordpunct_tokenize
from nltk.tokenize.treebank import TreebankWordDetokenizer
from googletrans import Translator
from TurkishStemmer import TurkishStemmer

path = "SECOND_EDITION/"

clearTxtPath = (path + "/CleanedAllTogether/")

selected_languages = ['english','russian','spanish',
                      'german','french','portuguese','turkish']

nltk_stopwords_languages = ['arabic', 'azerbaijani', 'danish', 'dutch',
                            'english', 'finnish', 'french', 'german', 'greek',
                            'hungarian', 'indonesian', 'italian', 'kazakh',
                            'nepali', 'norwegian', 'portuguese', 'romanian',
                            'russian', 'slovene', 'spanish','swedish', 'tajik',
                            'turkish']

"""
#-------------------------- Translate2English --------------------------#
"""
def func_translate2english(text,language):
    translator = Translator(service_urls=None,
                            user_agent='Mozilla/5.0 (Windows NT 10.0; Win64; x64)',
                            proxies=None, timeout=None)
    if 0.7 < translator.detect(text).confidence:
        translation = translator.translate(text, dest='en').text
    else:
        translation = translator.translate(text,src = language, dest='en').text
    return translation

    """
    #-------------------------- String2Token ----------------------------#
    """
def func_string2token(text):
    tokenized_word=word_tokenize(text)
    return tokenized_word

    """
    #-------------------------- HTML2Text ------------------------------#
    """
def func_htmlTotext(html):
```

```
    h = html2text.HTML2Text()
    h.ignore_links = True
    h.bypass_tables = False
    html_removed=h.handle(html)
    return html_removed

    """
    #-------------------------- Stemmer selection ----------------------#
    """

def func_stem(text,stem,language):
    if stem == "lemmatization":
        stemText=func_lemmaSentence(text,language)
    if stem == "snowball":
        stemText=func_snowballStemSentence(text,language)
    return stemText

    """
    #-------------------------- Stemming -------------------------------#
    """
def func_snowballStemSentence(text,language):
    if language == "turkish":
        ps = TurkishStemmer()
    elif language == "english":
        ps = SnowballStemmer(language)
    elif language == "russian":
        ps = SnowballStemmer(language)
    elif language == "spanish":
        ps = SnowballStemmer(language)
    elif language == "german":
        ps = SnowballStemmer(language)
    elif language == "french":
        ps = SnowballStemmer(language)
    elif language == "portuguese":
        ps = SnowballStemmer(language)
    token_words=word_tokenize(text)
    token_words
    stem_sentence=[]
    for word in token_words:
        stem_sentence.append(ps.stem(word))
        stem_sentence.append(" ")
    return "".join(stem_sentence)

            """
            #-------------------------- Lemmatization --------------------#
            """

       def func_lemmaSentence(text,language):
            lm = WordNetLemmatizer()
            token_words=word_tokenize(text)
            token_words
            lem_sentence=[]
            for word in token_words:
                lem_sentence.append(lm.lemmatize(word))
                lem_sentence.append(" ")
            return "".join(lem_sentence)

            """
            #-------------------------- Remove chars ---------------------#
            """

       def func_remove_chars(text,chars):
            if chars == "allChars":
                clearChars = re.sub("[^a-zA-
   Z ĞÜŞİÖÇğüşiöçIı]+", "", str(text))
            #nokta, virgül, tek tırnak ve noktalı virgül hariç
```

```
123.            elif chars == "exCommaDotSQM":
124.                clearChars = re.sub("[^a-zA-
    Z ĞÜŞİÖÇğüşiöçIı.,';âáàêéèëîíìïôóòûúùæ]+", "", str(text))
125.            #3 harften kısa harf grupları
126.            less3 = re.sub(r'\W*\b\w{1,3}\b', ' ', clearChars).strip()
127.            #20 harften uzun harf grupları
128.            more20 = re.sub(r'\w{20,100}', '', less3).strip()
129.            # büyük harf küçük harf dönüşümü
130.            lowerCase=more20.lower()
131.            #çoklu boşlukların silinmesi
132.            clearedChars=re.sub(' +', ' ', lowerCase)
133.            return clearedChars
134.
135.            """
136.            #-------------------------- Remove stop_words ----------------#
137.            """
138.        def func_remove_stop_words(text):
139.            stop_words = set(stopwords.words('english'))
140.            word_tokens = word_tokenize(text)
141.            removed = [w for w in word_tokens if not w in stop_words]
142.            stop_words_removed = TreebankWordDetokenizer().detokenize(remove
    d)
143.            return stop_words_removed
144.
145.            """
146.            #-------------------------- Remove non-english ---------------#
147.            """
148.        def func_remove_non_english(not_pure_english):
149.            """ import words from nltk 236737 words"""
150.            words = set(nltk.corpus.words.words())
151.
152.            english = " ".join(w for w in nltk.wordpunct_tokenize(not_pure_e
    nglish) if w.lower() in words or not w.isalpha())
153.            return english
154.
155.            """
156.            #-------------------------- Detect Language ------------------#
157.            """
158.        def _calculate_languages_ratios(text):
159.
160.            languages_ratios = {}
161.            tokens = wordpunct_tokenize(text)
162.            words = [word.lower() for word in tokens]
163.
164.            """
165.            stopwords.fileids()
166.            ['arabic', 'azerbaijani', 'danish', 'dutch', 'english', 'finnish
    ', 'french',
167.             'german', 'greek', 'hungarian', 'indonesian', 'italian', 'kazak
    h', 'nepali',
168.             'norwegian', 'portuguese', 'romanian', 'russian', 'slovene', 's
    panish',
169.             'swedish', 'tajik', 'turkish']"
170.            """
171.            #Metinde görünen benzersiz stopwords sayısının her dil için hesa
    plaması
172.            for language in stopwords.fileids():
173.                stopwords_set = set(stopwords.words(language))
174.                words_set = set(words)
175.                common_elements = words_set.intersection(stopwords_set)
176.
177.                languages_ratios[language] = len(common_elements)
178.
179.            return languages_ratios
180.            """
181.            #-----------------------------------------------------------#
```

```
            """
        def func_detect_language(text):
            ratios = _calculate_languages_ratios(text)
            most_rated_language = max(ratios, key=ratios.get)
            return most_rated_language

            """
            #-------------------------- Save output ----------------------#
            """
        def func_save_output(text,language):
            finalString = (line+","+language+","+text)
            outfilename = (line+","+language+".txt")
            # save output with url string as filename
            filename= (clearTxtPath + str(outfilename))
            with open(filename, 'w') as f:
                print('',finalString, file=f)  # Python 3.x

            """
            #-------------------------- Main Task ------------------------#
            """

        with open(path+'smalldataset/txtFileList') as f:
            lines = f.readlines()
            for line in lines:
                line = line.rstrip()
                print(line)
                txtPath = path+"smalldataset/htmlRawText6208/"+line+".txt"
                with open(txtPath,'r') as file:
                    html = file.read().replace('\n', '')
                    #line="Advertising_and_Marketing,asyapromosyon.com"
                    #------ Step 1 -----
                    text1 = func_htmlTotext(html)
                    #------ Step 2 -----
                    #text2 = string2token(text)
                    #------ Step 3 -----
                    language = func_detect_language(text1)
                    #------ Step 4 -----
                    if not language in selected_languages and language in nl
    tk_stopwords_languages:
                        #------ Step 5 -----
                        func_save_output(text1,language)
                    elif not language in nltk_stopwords_languages:
                        func_save_output(text1,"unknown")
                    else:
                        text5 = func_remove_chars(text1,"exCommaDotSQM")
                        #lemmatization,snowball
                        if not language == "english":
                            #------ Step 7 -----
                            text5 = func_translate2english(text5,language)
                            #if any(x.isupper() for x in text7):
                        #------ Step 6 -----
                        text6 = func_stem(text5,"snowball","english")
                        #------ Step 8 -----
                        text8 = func_remove_chars(text6,"allChars")
                        #----- Step 10 -----
                        text10 = func_remove_non_english(text8)
                        #------ Step 11 -----
                        text11 = func_remove_stop_words(text10)
                        #------ Step 11 -----
                        func_save_output(text11,language)
```

EK B.3

```
features=["Advertising_and_Marketing","Alcohol","Business_and_Economy","Computer_and_Internet",
          "Education","Gambling","Games","Government_and_Politics","Health","Music","New_and_Media",
          "Religion","Sexual_Content","Social_Networking","Travel_and_Tourism","Video_and_Movies","Violence"]

"""
Sınıflandırma modeli, veri seti, vektörleştirme metodu, normalleştirme bilgilerini parametre olarak alan fonksiyon
"""
def classification(classifier, dataset, vectorizer, normalization, confMatrixDetail) :
    if dataset == "wisimo":
        df = pd.read_csv('wisimo.csv', delimiter=',',header=None)
        df = df.replace(np.nan, 'home', regex=True)
    elif dataset == "porterStemmer" :
        df = pd.read_csv('wisimo_porterStemmer.csv', delimiter=',',header=None)
    elif dataset == "lancasterStemmer" :
        df = pd.read_csv('wisimo_lancasterStemmer.csv', delimiter=', ',header=None)
    elif dataset == "lemmatization" :
        df = pd.read_csv('wisimo_Lemmatization.csv', delimiter=',',header=None)
    else:
        df = pd.read_csv('wisimo.csv', delimiter=',', header=None)
        df = df.replace(np.nan, 'home', regex=True)

    with open('WebCategories') as f:
        lines = f.readlines()
        #lines= [x.strip() for i in lines]
        for line in lines:
            line = line.strip('\n')
            #line="Sports"
            print("Number of category %s web pages:" % line,df[df[0] == line][0].count())

    """
        Eğitim ve test veri setlerinin hazırlanması
    """
    X_train_raw, X_test_raw, y_train, y_test = train_test_split(df[2],df[0],test_size=0.3, random_state=42)
    print("Size of train set {0} ".format(X_train_raw.shape))
    print("Size of test set {0} ".format(X_test_raw.shape))
    #X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.25)

    """
        Vektörleştirme
    """
    if vectorizer == "count" :
        vectorizer = CountVectorizer(ngram_range=(1, 3),stop_words='english')
    elif vectorizer == "tfidf" :
        vectorizer = TfidfVectorizer(max_features=1500, min_df=5, max_df=0.7, stop_words=stopwords.words('language'))
    else :
        vectorizer = CountVectorizer(ngram_range=(1, 3),stop_words='english')
```

```
47.     X_train = vectorizer.fit_transform(X_train_raw)
48.     X_test = vectorizer.transform(X_test_raw)
49.
50.     """
51.         Normalleştirme
52.     """
53.     if normalization == "normalization" :
54.         from sklearn import preprocessing
55.         X_train = preprocessing.normalize(X_train)
56.         X_test = preprocessing.normalize(X_test)
57.
58.     """
59.     Sınıflandırma lgoritmaları ile model eğitilmesi
60. Not: Bu kısmi kod yalnızca varsayılan parametreler ile modellerin
    eğitilmesini gösterir. Çalışma kapsamında çapraz doğrulama ile elde edilen
    optimal parametreler kullanımıştır.
61.     """
62.     if classifier == "logistic" :
63.         clf = LogisticRegression()
64.         clf.fit(X_train , y_train)
65.         y_pred = clf.predict(X_test)
66.         print(classification_report(y_test, y_pred))
67.         print('Accuracy score: ',round(accuracy_score(y_pred,y_test),4))
68.     elif classifier == "svm" :
69.         svclassifier = SVC(kernel='linear')
70.         svclassifier.fit(X_train, y_train)
71.         print(classification_report(y_test, y_pred))
72.         print('Accuracy score: ',round(accuracy_score(y_pred,y_test),4))
73.     elif classifier == "decisiontree" :
74.         decision_tree_classifier = DecisionTreeClassifier()
75.         decision_tree_classifier.fit(X_train, y_train)
76.         y_pred = decision_tree_classifier.predict(X_test)
77.         print(classification_report(y_test, y_pred))
78.         print('Accuracy score: ',round(accuracy_score(y_pred,y_test),4))
79.     elif classifier == "randomforest" :
80.         clf = RandomForestClassifier()
81.         clf.fit(X_train  , y_train)
82.         y_pred = clf.predict(X_test)
83.         print(classification_report(y_test, y_pred))
84.         print(round(accuracy_score(y_pred,y_test),4))
85.     elif classifier == "neurolnetwork" :
86.         clf = MLPClassifier(solver='lbfgs', alpha=1e-5,
87.                             hidden_layer_sizes=(5, 2), random_state=1)
88.         clf.fit(X_train, y_train)
89.         y_pred = clf.predict(X_test)
90.         print(metrics.f1_score(y_test, y_pred, average='weighted', labels=n
    p.unique(y_pred)))
91.         print(classification_report(y_test, y_pred))
92.         print(accuracy_score(y_pred,y_test))
93.     elif classifier == "svmovr" :
94.         clf = OneVsRestClassifier(SVC(kernel='linear'))
95.         clf.fit(X_train, y_train)
96.         y_pred = clf.predict(X_test)
97.         print(classification_report(y_test, y_pred))
98.         print(round(accuracy_score(y_pred,y_test),4))
99.     elif classifier == "knn" :
100.             kNN = KNeighborsClassifier(n_neighbors=1)
101.             kNN.fit(X_train, y_train)
102.             y_pred=kNN.predict(X_test)
103.             print(classification_report(y_test, y_pred))
104.             print('Accuracy score: ',round(accuracy_score(y_pred,y_test)
    ,4))
105.         elif classifier == "bayes" :
106.             naive_bayes = MultinomialNB()
107.             naive_bayes.fit(X_train , y_train)
108.             y_pred = naive_bayes.predict(X_test)
```

```
109.                print('Precision score: ', precision_score(y_test, y_pred,av
    erage=None))
110.                print('Recall score: ', recall_score(y_test, y_pred,average=
    'micro'))
111.                print(classification_report(y_test, y_pred))
112.                print('Accuracy score: ',round(accuracy_score(y_pred,y_test)
    ,4))
113.
114.            elif classifier == "ensemble-gradient" :
115.                GradientBoosting = GradientBoostingClassifier(n_estimators=1
    00, learning_rate=1.0,
116.                                                              max_depth=1, r
    andom_state=0).fit(X_train, y_train)
117.                GradientBoosting.score(X_test, y_test)
118.                y_pred = GradientBoosting.predict(X_test)
119.                print(classification_report(y_test, y_pred))
120.                print(round(accuracy_score(y_pred,y_test),4))
121.
122.            elif classifier == "ensemble-bagging" :
123.                bagging = BaggingClassifier(KNeighborsClassifier(),
124.                                            max_samples=0.5, max_features=0.
    5).fit(X_train, y_train)
125.                bagging.score(X_test, y_test)
126.                y_pred = bagging.predict(X_test)
127.                print(classification_report(y_test, y_pred))
128.                print(round(accuracy_score(y_pred,y_test),3))
129.
130.            elif classifier == "ensemble-extratree" :
131.
132.                extraTrees = ExtraTreesClassifier(n_estimators=10, max_depth
    =None,
133.                                                  min_samples_split=2, rando
    m_state=0)
134.                scores = cross_val_score(extraTrees, X_train, y_train, cv=5)

135.                scores.mean()
136.                extraTrees.fit(X_train, y_train)
137.                y_pred = extraTrees.predict(X_test)
138.                print(classification_report(y_test, y_pred))
139.                print(round(accuracy_score(y_pred,y_test),4))
140.
141.            elif classifier == "ensemble-adaboost" :
142.                adaBoost = AdaBoostClassifier(n_estimators=100, random_state
    =0)
143.                adaBoost.fit(X_train, y_train)
144.                y_pred = adaBoost.predict(X_test)
145.                print(classification_report(y_test, y_pred))
146.                print(round(accuracy_score(y_pred,y_test),4))
147.
148.        classification("logistic","wisimo","count","yes","detailed")
```

EK C.1

Çizelge C.1 : Sınıflandırma algoritmaları optimal parametreleri

Sınıflandırma Algoritmaları	Optimal Parametreler
Logistic Regression	'C': 100, fit_intercept: True, multi_class: ovr, n_jobs: 1, solver: liblinear
Destek Vektör Makineleri	'C': 10, 'gamma': 0.0001, 'kernel': ‘rbf'
OneVSRest	‘C’: 1, kernel: rbf
Rassal Orman	max_depth: 50, min_samples_leaf: 2, min_samples_split: 3, n_estimators: 1000
Karar Ağaçları	max_depth=None, min_samples_split=2
k-En Yakın Komşuluk	n_neighbors: ‘7’, weights: 'distance'
Kolektif ExtraTrees	n_estimators=10, max_depth=None, ... min_samples_split=2, random_state=0
Kolektif BaggingClassifier	max_samples: 0.5, max_features: 0.5
Kolektif GradientBoosting	n_estimators: 16, learning_rate: 0.8

EK C.2

max_depth: Bir ağacın maksimum derinliğini belirler.

min_samples_split: Bölünmeyi göz önüne almak için bir düğümün içermesi gereken minimum örnek sayısı.

min_samples_leaf: Bir yaprak düğümü olarak kabul edilmesi gereken minimum örnek sayısı. Ağacın büyümesini sınırlamak için bu parametreyi kullanın.

max_features: En iyi bölünmeyi ararken göz önünde bulundurulması gereken özelliklerin sayısı.

n_estimators: Ormandaki ağaç sayısı

kernel: Algoritmada kullanılacak çekirdek türünü belirtir.

Gamma: çekirdek katsayısnı belirtir.

C: Düzenleme parametresi. Gücü büyüklüğü ile ters orantılıdır.

weight: Tahminde kullanılan ağırlık fonksiyonudur. Uzaklıklar ile etkiler arasındaki ilişkiyi tanımlar.

n_neighbors: Komşuluk sorguları için kullanılacak komşu sayısı.

ÖZGEÇMİŞ

Ad Soyad **:** Kenan Enes Aydın

Doğum Yeri ve Tarihi **:** Üsküdar, 23 Mayıs 1990

E-Posta **:** aydinke17@itu.edu.tr

EĞİTİM

- **Lisans:** Anadolu Üniversitesi, İşletme Fakültesi, İşletme.
- **Lisans:** Uludağ Üniversitesi, Mühendislik Fakültesi, Elektrik-Elektronik Mühendisliği

PROFESYONEL DENEYİM:

- Lova Ağ Teknolojileri, Yazılım Mühendisi, (2016-2018)
- Ecemtag Kontrol Teknolojieri, Gömülü Yazılım Mühendisi (2018- Halen)