

**T. C.**

**SELÇUK ÜNİVERSİTESİ**

**TEKNOLOJİ FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**VERİ MADENCİLİĞİ DERSİ**

**EMLAK VERİSİ**

**213301043 – ibrahim YAŞAR**

**213301044 – Muhammed Emre NUROĞLU**

**213301012 – Yusuf İKRİ**

**Doç. Dr. Murat KÖKLÜ**

**11 Ocak 2024**

**İÇİNDEKİLER**

[1. ÖZET 3](#_Toc154794321)

[2. ÖDEVDE KULLANACAĞINIZ PROGRAMIN ANLATIMI 4](#_Toc154794322)

[3. SINIFLANDIRMA ALGORİTMALARI 5](#_Toc154794323)

[3.1. K-en Yakın Komşu(KNN) 5](#_Toc154794324)

[3.2. Karar Ağacı(Desicion Tree) 5](#_Toc154794325)

[3.3. Random Forest 6](#_Toc154794326)

[4. MATERYAL VE YÖNTEM 6](#_Toc154794327)

[4.1. Verinin Elde Edilmesi 6](#_Toc154794328)

[4.2. Verinin Özellikleri 8](#_Toc154794329)

[4.3. Performans Ölçümleri 9](#_Toc154794330)

[4.4. Çapraz Doğrulama 10](#_Toc154794331)

[5. ARAŞTIRMA SONUÇLARI VE TARTIŞMA 12](#_Toc154794332)

[5.1. KNN Algoritması Sonuçları 12](#_Toc154794333)

[5.2. Decision Tree Sonuçları 14](#_Toc154794334)

[5.3. Random Forest Algoritması Sonuçları 17](#_Toc154794335)

[6. SONUÇLAR 19](#_Toc154794336)

[7. ÖNERİLER 22](#_Toc154794337)

[8. KAYNAKLAR 23](#_Toc154794338)

# 1. ÖZET

Bu çalışmamızda bir emlak sitesinden çekmiş olduğumuz verileri kullanarak evlerin fiyatlarına göre gruplandırma yapmak amacıyla çeşitli yapay zekâ algoritmaları ile bir makine öğrenimi modeli geliştirmeyi amaçladık. Öncelikle çeşitli ev özelliklerini ve fiyatlarını içeren veri setini Python ile elde ettik. Bu verileri bir emlak sitesinden request kütüphanesiyle toplanmış ve ardından beatifulsoup kütüphanesiyle düzenlenerek kaydedilmiştir. Bu kaydedilen verilerin tam, gürültülü(parazitli) ve tutarlı olmamasından dolayı veriler veri ön işlemesinden geçirilerek yapay zekâ modelinde eğitilebilir bir hale getirilmiştir. Temizlenen verileri fiyatına göre uygun gruplandırma algoritmaları kullanılarak gruplandırma işlemi yapılmıştır. GradientBoostingRegressor ve XGBRegressor algoritmaları, bu veri seti üzerinden eğitilmiştir. Eğitim sürecinde verilerin %80 ’i model öğrenmesi geri kalan %20 ’si de doğruluk testi için kullanıldı.

Bu eğitim sonuçlarında algoritmalarda sırasıyla %77 ve %71 doğruluk oranları elde edilmiştir. Bu eğitilen modelleri kullanarak kullanıcılardan gerekli ev özellikleri alınarak evin özelliğine göre daha önceden belirlenmiş fiyat gruplardan hangisine uygun olduğunu belirlenmektir. Bu gruplandırmada elde edilen sonuçlar kullanıcıların ev alım ve satımlarında daha bilinçli ve mantıklı kararlar vermesinde yardımcı olmayı amaçlamıştır.

Projenin doğruluk oranı ve daha doğru sonuçlar vermesi için veri setindeki özellik sayısı ve veri sayısı arttırılarak daha doğru ve keskin sonuçlar vermesi sağlanabilir.

# 

# 2. ÖDEVDE KULLANACAĞINIZ PROGRAMIN ANLATIMI

Bu ödevde, veri analizi ve makine öğrenimi için Python programlama dilini kullanacağım. Python, geniş bir kütüphane ekosistemi ve kolay okunabilir syntax yapısıyla bilinir. Proje sürecinde çeşitli kullandık. İşte kullandığımız kütüphaneler ve genel kullanımımız.

**Python Kütüphaneleri**

Scikit-learn: Makine öğrenimi için geniş bir araç seti sunan bu kütüphane, sınıflandırma, regresyon, kümeleme gibi birçok algoritma içerir.[1]

Pandas: Veri analizi ve manipülasyonu için kullanılan bu kütüphane, veri setlerini daha etkili bir şekilde işlememizi sağlar.[2]

Matplotlib: Veri görselleştirmesi için kullanılan bu kütüphane, grafikler ve çizimler oluşturarak sonuçları daha anlaşılır hale getirir.[3]

BeautifulSoup (bs4): HTML ve XML dosyalarını ayrıştırmak ve içeriklerini çekmek için kullanılan bir kütüphanedir. [4]

Requests: Web sayfalarına HTTP talepleri göndermek için kullanılır ve böylece veri çekme işlemleri gerçekleştirilir.[5]

**Genel Kullanım:**

Veri Çekme: BeautifulSoup ve requests kütüphaneleri ile web sayfasından projede kullanıcağımız veriler çekilir.

Veri Analizi: Pandas kütüphanesi ile çekilen verileri DataFrame'e dönüştürüldü ve analiz etmek için kullanıldı.

Veri Hazırlığı: İlk olarak, projede kullanılacak veri seti Pandas kütüphanesi ile yüklenir ve temizlenir. Eksik veriler doldurulur veya çıkartılır.

Veri Görselleştirmesi: Matplotlib kullanılarak, veri setinin özellikleri ve dağılımları görselleştirilir. Bu adım, veri üzerindeki genel yapıyı anlamamıza yardımcı olur.

Veri Temizleme ve Hazırlık: NumPy kullanarak veri setini temizledi ve uygun model ile eğitim ve test setlerine ayrılır.

Makine Öğrenimi Modeli Oluşturma: Scikit-learn kütüphanesi ile bir uygun algoritma seçilir. Model eğitilir ve veri seti üzerinde test edilir.

Sonuçların Değerlendirilmesi: Modelin performansı, çeşitli metriklerle (doğruluk, hassasiyet, recall vb.) değerlendirilir. Bu adım, modelin ne kadar başarılı olduğunu anlamamıza yardımcı olur.

# 3. SINIFLANDIRMA ALGORİTMALARI

Sınıflandırma ayrık değişkenler için öngörüde (prediction) bulunmak, sınıf niteliğini belirlemek için diğer nitelikleri kullanarak bir model bulmaktır. Amacı ise yeni bir kayıt geldiğinde, bu kaydı geliştirilen modeli kullanılarak mümkün olduğunca doğru bir sınıfa atamaktır.

## 3.1. K-en Yakın Komşu (KNN)

Gözetimli öğrenme algoritmalarından biridir. Sınıflandırma ve regresyon problemlerinde kullanılabilir. KNN, bir yeni verinin sınıfını veya değerini, o veriye en yakın olan K sayıdaki komşusunun sınıfına veya değerine göre tahmin eder.

KNN, parametrik olmayan bir algoritmadır. Bu, modelin eğitim verisi üzerinde öğrenmediği ve yeni veriler için tahmin yaparken herhangi bir varsayım yapmadığı anlamına gelir. KNN, basit ve hızlı bir algoritmadır. Ancak, yüksek boyutlu veriler için performansı düşük olabilir.

KNN'nin avantajları şunlardır:

* Basit: KNN, öğrenmesi ve uygulaması kolay bir algoritmadır.
* Hızlı: KNN, yeni veriler için hızlı tahminler yapabilir.
* Genel amaçlı: KNN, sınıflandırma ve regresyon gibi farklı problemlerde kullanılabilir.

KNN'nin dezavantajları şunlardır:

* Yüksek boyutlu veriler için performansı düşük olabilir.
* Aşırı öğrenmeye (overfitting) eğilimli olabilir. [6]

## 3.2. Karar Ağacı (Desicion Tree)

Gözetimli öğrenme algoritmalarından biridir. Sınıflandırma ve regresyon problemlerinde kullanılabilir. Karar ağacı, bir yeni verinin sınıfını veya değerini, o veriye bir dizi soru sorarak tahmin eden bir algoritmadır.

Karar ağacı, eğitim verilerini bir dizi soru sorarak analiz eder. Her soru, verinin belirli bir özelliğine odaklanır. Sorunun yanıtı, ağacın sonraki dalına yönlendirir. Bu süreç, yeni veri, ağacın en alt dalına ulaşana kadar devam eder. Bu dal, yeni verinin sınıfını veya değerini temsil eder.

Karar ağacı, basit ve anlaşılması kolay bir algoritmadır. Bu nedenle, yeni başlayanlar için iyi bir seçimdir. Ayrıca, karmaşık verilerle iyi başa çıkabilir.[7]

## 3.3. Random Forest

Sınıflandırma ve regresyon için kullanılan bir ensemble öğrenme algoritmasıdır. Bu algoritma, çok sayıda karar ağacı oluşturur ve tahminlerini bu ağaçların çıktılarını birleştirerek yapar.

Random forest, aşırı öğrenmeye (overfitting) karşı dirençli olması ve genellikle yüksek doğruluk sağlaması nedeniyle popüler bir algoritmadır.

Random forest, öğrenme verisi kümesinden rastgele seçilen özellikler kullanılarak karar ağaçları oluşturulur. Bu, ağaçların birbirinden farklı özelliklerle eğitilmesini sağlar ve aşırı öğrenmeye karşı direncini artırır.

Random forest, bir yeni verinin sınıfını veya değerini tahmin etmek için, bu verinin her karar ağacı tarafından tahmin edilen sınıflarını veya değerlerini birleştirir. Ağaçların tahminleri ne kadar benzerse, yeni verinin tahmin edilen sınıfı veya değeri de o kadar kesin olur.

Random forest, sınıflandırma ve regresyon için kullanılabilecek çeşitli parametrelere sahiptir. Bu parametreleri ayarlayarak, modelin performansını iyileştirebilirsiniz [8]

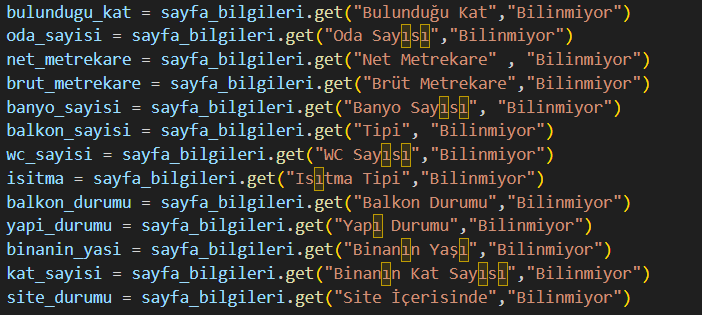
# 4. MATERYAL VE YÖNTEM

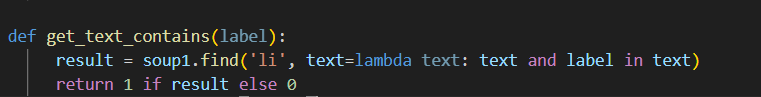
## 4.1. Verinin Elde Edilmesi

Projemizin amacı ev özelliklerinden fiyatlarına göre gruplandırarak sonuç vermesidir. Projemizde kullanmış olduğumuz verileri emlakjet isimli emlak sitesinden çekmiş bulunmaktayız. Bu verileri çekerken Python programlama dilini ve pythonda bulunan BeautifulSoup, requests ve pandas kütüphanelerini kullandık.

Öncelikle requests kütüphanesini kullanarak emlakjetin istanbuldaki evlerinin olduğu site url sine bağlantı isteği gönderdik. Daha sonra bize gelen sonuç html formatında olduğu için BeautifulSoup ile bu gelen sonucu düzenlenmiş bir forma dönüştürdük. Bu düzenlenmiş formdan öncelikle alacağımız verilerin hangi etiket içinde bulunduğuna baktık. Bulduğumuz etikete göre almak istediğimiz veriler <div> </div> etiketi içinde ve div classının “\_3qUI9q” olduğunu öğrendik. Daha sonra bu etiket içinden sitenin içerisinde bulunan aradığımız ilanları for döngüsü kullanarak tek tek dolaşarak sırasıyla evlerin fiyatını, bulunduğu şehri, bulunduğu semtinin span etiketlerinde olduğunu öğrendik ve onları çektik. Ondan sonra detaylarını görebileceğimiz <a> etiketi içerisinde bulunan linkleri aldık.

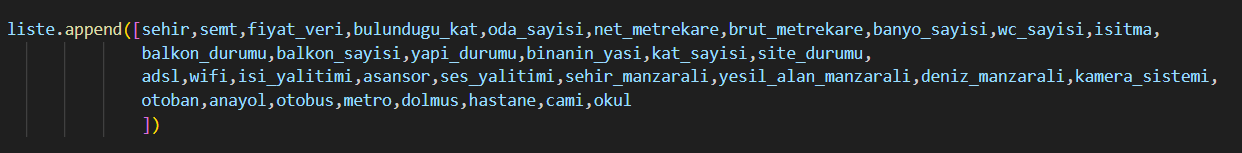
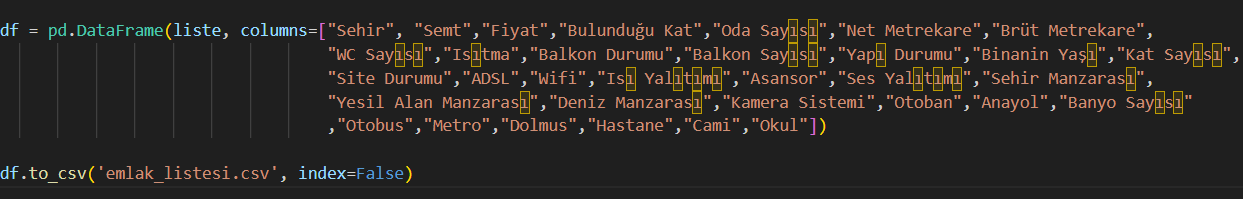
Aldığımız bu linklere requests kütüphanesi yardımıyla tekrardan istek atarak içerisinde bulunan verileri çekmeye başladık. Öncelikle ana verilerin bulunduğu classı “\_1bVOdb” olan divdeki kat, oda sayısı, net metrekare, brüt metrekare, banyo sayısı, balkon sayısı, ısıtma tipi, balkon durumu, yapı durumu, binanın yaşı, kat sayısı, site durumu özelliklerini yazdığımız özel fonksiyon ile çektik. Daha sonra ul altında li etiketi içerisinde olan detay verilerinide yazmış olduğumuz farklı bir fonksiyon ile almış olduk.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

En sonda almış olduğumuz tüm verileri tanımladığımız bir listeye attık. Daha sonra pandas kütüphanesini kullanarak listedeki verileri Dataframe formatına çevirdik. Çevirmiş olduğumuz Dataframe dosyasını en sonda csv dosyası olarak kaydettik. Bunun sonucunda 11019 satır 32 sütundan oluşan bir veri kümesi elde etmiş olduk.



## 4.2. Verinin Özellikleri

Elimizdeki verilerin şehir, semt, ısıtma, ev türü adında olan özelliklerimiz string ifade şeklindedir.

Fiyat, bulunduğu kat, net metrekare, brüt metrekare, banyo sayısı, wc sayısı, yapı durumu, kat sayısı adında olan özelliklerimiz integer ifade şeklindedir.

Balkon durumu, ADSL, wifi, ısı yalıtımı, asansör, ses yalıtımı, şehir manzaralı, deniz manzaralı, yeşil alan manzaralı, kamera sistemi, otoban, anayol, metro, dolmuş, hastane, cami, okul adında olan özelliklerimiz 1 veya 0 ifadesi şeklindedir. Oda sayısı adındaki özelliğimiz ise 3+1, 4+1 vb. şekildedir.

**Oda Sayısı**: 3+1, 4+1, 5+1 vb. ifadelerini 4, 5, 6 vb. yaptık

**Bulunduğu Kat**: Çatı katı yazan ifadelerine kat sayısındaki ifadeyi atadık, Giriş kat ve bahçe kat yazan ifadelere 0 değerini atadık.

**Semt:** Fatih, Arnavutköy, Ataşehir, Bağcılar, Bahçelievler, Bakırköy, Başakşehir, Bayrampaşa, Beşiktaş, Beylikdüzü, Büyükçekmece, Çekmeköy, Eyüp sultan, Gaziosmanpaşa, Güngören, Kadıköy, Kağıthane, Kartal, Üsküdar, Ümraniye, Maltepe, Şişli ifadelerine sırasıyla 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11,12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 ifadelerini atadık.

**Şehir:** Projemizde sadece İstanbul verilerini kullandığımızdan dolayı şehir isimli sütunu kaldırdık.

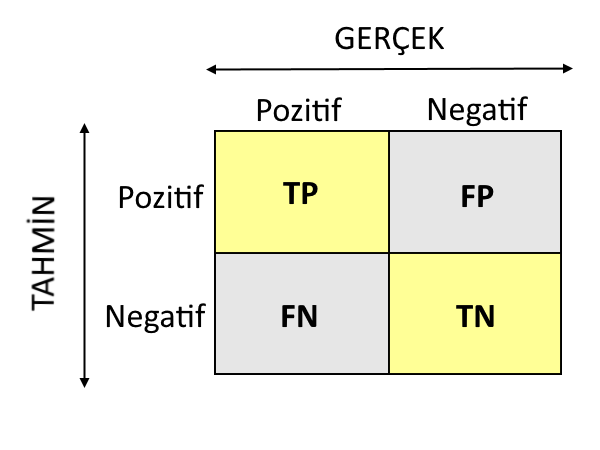
**Bina Türü:** Çektiğimiz verilerin hepsinde bina türü daire olduğu için kaldırdık.

**Isıtma:** Sobalı, klimalı, doğalgaz sobalı, yerden ısıtma, merkezi doğalgaz, merkezi (pay ölçer), kombi doğalgaz ifadelerine sırasıyla 1,2,3,4,5,6,7 ifadelerini atadık.

## 4.3. Performans Ölçümleri

***Karmaşıklık Matrisi:***

Veri Madenciliğinde sınıflandırma modellerinin performansını değerlendirmek için hedef niteliğe ait tahminlerin ve gerçek değerlerin karşılaştırıldığı matrise hata matrisi denir. Her ne olursa olsun sınıflandırma tahminleri şu dört değerlendirmeden birine sahip olacaktır:



* Gerçek Pozitif (TP): Gerçekte pozitif bir durumu ifade eden ve sınıflandırıcı tarafından pozitif olarak tahmin edilen örnekleri ifade eder.
* Gerçek Negatif (TN): Gerçekte negatif bir durumu ifade eden ve sınıflandırıcı tarafından negatif olarak tahmin edilen örnekleri ifade eder.
* Yanlış Pozitif (FP): Gerçekte negatif bir durumu ifade eden ve sınıflandırıcı tarafından pozitif olarak tahmin edilen örnekleri ifade eder.
* Yanlış Negatif (FN): Gerçekte pozitif bir durumu ifade eden ve sınıflandırıcı tarafından negatif olarak tahmin edilen örnekleri ifade eder.

**Ortalama Accuracy (%):**

Ortalama Accuracy, doğru tahminlerin toplam veri noktalarına oranını ölçer. Accuracy şu şekilde hesaplanır:

**Ortalama Precision (%):**

Precision, bir sınıfa ait olarak tahmin edilen örneklerin gerçekten o sınıfa ait olan örneklerin oranını ölçer. Precision şu şekilde hesaplanır:

**Ortalama Recall (%):**

Recall, gerçekte bir sınıfa ait olan örneklerin ne kadarının doğru bir şekilde tahmin edildiğini ölçer. Recall şu şekilde hesaplanır:

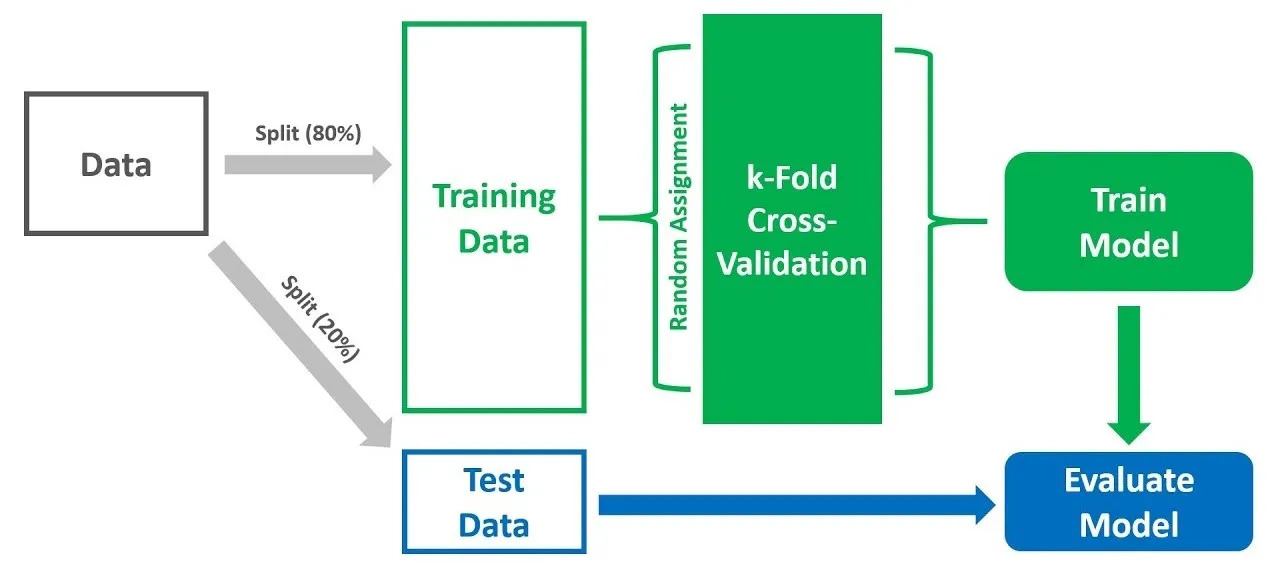
**Ortalama F1-Skor (%):**

F1-Skor, precision ve recall ‘in harmonik ortalamasıdır ve hem precision hem de recall'e önem veren bir metriktir. F1-Skor şu şekilde hesaplanır:

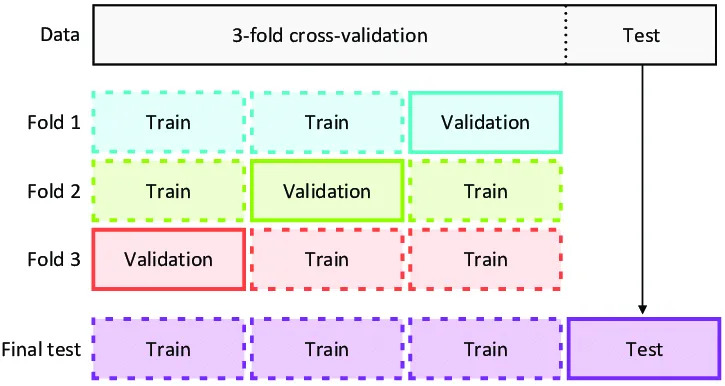
## 4.4. Çapraz Doğrulama

K-Fold Cross Validation, makine öğrenimi ve istatistiksel modelleme gibi veri analizi yöntemlerinde yaygın olarak kullanılan bir doğrulama tekniğidir. Temel amacı, bir modelin performansını değerlendirmek ve genelleme yeteneğini ölçmek için kullanılan veri setinin etkili bir şekilde kullanılmasını sağlamaktır.

Modelimizin performansını en iyi şekilde değerlendirebilmek için öncelikle veri setimizi test ve eğitim veri seti olarak belirli bir oranda ikiye ayırıyoruz. Bunu yapmamızdaki amaç, test setindeki verinin eğitim setindeki veriyi görüp öğrenmemesidir.



K-Fold Cross Validation, eğitim veri setini K adet eşit parçaya böler ve ardından bir parçayı Validation yani kendi içerisinde doğrulama yapmak için kullanırken diğer parçaları eğitim veri seti olarak kullanır. Bu işlem K defa tekrarlanır, her seferinde farklı bir parça validation verisi olarak seçilir. Sonuç olarak, K farklı doğruluk değerleri elde edilir ve bu değerlerin ortalaması alınarak modelin performansı değerlendirilir.



K sayısı, doğruluk tahminlerinin güvenilirliğini artırmak için yeterli büyüklükte olmalıdır. Ancak aynı zamanda hesaplama ve zaman maliyetlerini de dikkate almak önemlidir. K için zayıf bir değer, modelin ölçülen yeteneğinin olduğundan fazla tahmin edilmesine (yüksek yanlılık) veya kullanılan eğitim verilerine bağlı olarak büyük ölçüde değişmesine (yüksek varyans) neden olabilir.

Genellikle, K sayısı için yaygın olarak tercih edilen değerler 5 ve 10'dur. Ayrıca, veri setinin boyutu arttıkça K değerinin de artırılması önerilir. Örneğin, büyük veri setleri için 10 veya daha yüksek K değerleri tercih edilebilir. [9]

**5. ARAŞTIRMA SONUÇLARI VE TARTIŞMA**

## 5.1. KNN Algoritması Sonuçları

TAHMİNİ SINIF

GERÇEK

SINIF

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Sınıf 1 | Sınıf 2 | Sınıf 3 | Sınıf 4 | Sınıf 5 |
| Sınıf 1 | 307 | 80 | 28 | 7 | 3 |
| Sınıf 2 | 167 | 159 | 72 | 21 | 5 |
| Sınıf 3 | 106 | 106 | 116 | 70 | 21 |
| Sınıf 4 | 54 | 79 | 84 | 118 | 72 |
| Sınıf 5 | 34 | 42 | 59 | 55 | 228 |

**Tablo 5.1.1** Knn algoritmasının hata matrisi

**Genel Model Performansı:**

Toplam 2093 test örneği üzerinde modelin doğruluk değeri %44 olarak elde edilmiştir. Bu, toplamda doğru sınıflandırılan örneklerin oranını gösterir.

**Sınıf Bazında Analiz:**

Sınıf bazında precision, recall ve f1-score değerleri incelenmiştir. Örneğin, sınıf 1 için precision %46, recall %72 ve f1-score 0,56 olarak bulunmuştur. Bu değerler, sınıf 1'in model tarafından ne kadar iyi tahmin edildiğini gösterir**.**

**Dengesiz Sınıflar:**

Sınıflar arasında dengesizlik gözlemlenmiştir. Özellikle, sınıf 5'in recall değeri diğer sınıflara göre daha yüksektir, bu da modelin sınıf 5'i daha iyi tanıdığını gösterir.

**Düşük Performanslı Sınıflar:**

Sınıf 3, sınıf 4 gibi düşük performanslı sınıfların düşük precision, recall ve f1-score değerleri vardır. Bu sınıfların model tarafından zayıf bir şekilde tahmin edildiği anlaşılmaktadır.

**Genel Değerlendirme:**

Weighted avg (ağırlıklı ortalama) değerlendirmesine göre, modelin genel performansının arttığı görülmektedir. Bu, modelin sınıfları daha dengeli bir şekilde tahmin ettiğini ve bazı sınıflarda önceki değerlere kıyasla daha iyi bir başarı elde ettiğini gösterir.

**İyileştirme Alanları:**

Modelin performansını daha da artırmak için özellikle düşük performanslı sınıflarda iyileştirmeler yapılması gerekmektedir. Bu, belirli sınıfların daha fazla örnek ile eğitilmesi, model parametrelerinin ayarlanması veya farklı bir modelin denemesi gibi stratejileri içerebilir.

|  |  |
| --- | --- |
| Performans Sonuçları | Yüzde (%) |
| Ortalama Accuracy | 44 |
| Ortalama Precesion | 45 |
| Ortalama Recall | 44 |
| Ortalama F1-Skor | 44 |

**Tablo 5.1.2** KNN algoritması performans sonuçları

**Ortalama Accuracy (%):**

Modelin, 5 farklı sınıfı doğru bir şekilde sınıflandırma oranı %44'tür. Bu değer, toplam örneklerin neredeyse yarısının doğru bir şekilde tahmin edildiğini göstermektedir.

**Ortalama Precision (%):**

Ortalama precision değeri %45'tir. Bu, modelin sınıfları doğru bir şekilde tahmin etme yeteneğinin orta seviyede olduğunu gösterir. Precision, bir sınıfın tahminlerinin ne kadarının gerçekten doğru olduğunu ölçer.

**Ortalama Recall (%):**

Ortalama recall değeri %44'tür. Bu, modelin gerçek pozitif örneklerin yaklaşık olarak yarısını yakalayabildiğini gösterir. Düşük bir recall değeri, özellikle sınıfların tanınma oranının orta seviyede olduğunu gösterir.

**Ortalama F1-Skor (%):**

Ortalama F1-Skor değeri %44'tür. F1-Skor, precision ve recall değerlerinin harmonik ortalamasını temsil eder. Bu değer, modelin sınıfları hem doğru hem de kapsamlı bir şekilde tahmin etmede orta seviyede başarılı olduğunu gösterir.

Sonuç olarak, modelin genel performansı orta seviyede bir doğruluk, precision, recall ve F1-Skor ile karakterize edilmektedir. Ancak, iyileştirme adımları, belirli sınıfların performansını artırmak ve modelin genel etkinliğini optimize etmek amacıyla gerçekleştirilebilir.

## 5.2. Decision Tree Sonuçları

TAHMİNİ SINIF

GERÇEK

SINIF

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Sınıf 1 | Sınıf 2 | Sınıf 3 | Sınıf 4 | Sınıf 5 |
| Sınıf 1 | 309 | 121 | 48 | 23 | 7 |
| Sınıf 2 | 134 | 220 | 111 | 49 | 14 |
| Sınıf 3 | 61 | 113 | 196 | 121 | 38 |
| Sınıf 4 | 26 | 66 | 120 | 203 | 106 |
| Sınıf 5 | 10 | 28 | 41 | 117 | 335 |

**Tablo 5.2.1** Decision Tree algoritmasının hata matrisi

**Genel Model Performansı:**

Toplam 2617 test örneği üzerinde modelin accuracy (doğruluk) değeri %48 olarak elde edilmiştir. Bu, toplamda doğru sınıflandırılan örneklerin oranını gösterir.

**Sınıf Bazında Analiz:**

Sınıfların precision, recall ve f1-score değerleri incelenmiştir. Örneğin, sınıf 1 için precision %57, recall %61 ve f1-score 0,59 olarak bulunmuştur. Bu değerler, sınıf 1'in model tarafından ne kadar iyi tahmin edildiğini gösterir.

**Dengesiz Sınıflar:**

Sınıflar arasında dengesizlik gözlemlenmiştir. Özellikle, sınıf 5'in recall değeri diğer sınıflara göre daha yüksektir, bu da modelin sınıf 5'i daha iyi tanıdığını gösterir.

**Düşük Performanslı Sınıflar:**

Sınıf 4, sınıf 5 gibi düşük performanslı sınıfların düşük precision, recall ve f1-score değerleri vardır. Bu sınıfların model tarafından zayıf bir şekilde tahmin edildiği anlaşılmaktadır.

**Genel Değerlendirme:**

Weighted avg (ağırlıklı ortalama) değerlendirmesine göre, modelin genel performansının orta seviyede olduğu görülmektedir. Ancak, iyileştirme alanları bulunmaktadır, özellikle düşük performanslı sınıflarda performansı artırmak önemlidir.

**İyileştirme Alanları:**

Modelin performansını artırmak için özellikle düşük performanslı sınıflarda iyileştirmeler yapılması gerekmektedir. Bu, belirli sınıfların daha fazla örnek ile eğitilmesi, model parametrelerinin ayarlanması veya farklı bir modelin denemesi gibi stratejileri içerebilir.

|  |  |
| --- | --- |
| Performans Sonuçları | Yüzde (%) |
| Ortalama Accuary | 48 |
| Ortalama Precesion | 48 |
| Ortalama Recall | 48 |
| Ortalama F1-Skor | 48 |

**Tablo 5.2.2** Decision Tree algoritmasının Performans Sonuçları

**Ortalama Accuracy (%):**

Modelin, 5 farklı sınıfı doğru bir şekilde sınıflandırma oranı %48'dir. Bu değer, toplam örneklerin neredeyse yarısının doğru bir şekilde tahmin edildiğini göstermektedir.

**Ortalama Precision (%):**

Ortalama precision değeri %48'dir. Bu, modelin sınıfları doğru bir şekilde tahmin etme yeteneğinin orta seviyede olduğunu gösterir. Precision, bir sınıfın tahminlerinin ne kadarının gerçekten doğru olduğunu ölçer.

**Ortalama Recall (%):**

Ortalama recall değeri %48'dir. Bu, modelin gerçek pozitif örneklerin yaklaşık olarak yarısını yakalayabildiğini gösterir. Düşük bir recall değeri, özellikle sınıfların tanınma oranının orta seviyede olduğunu gösterir.

**Ortalama F1-Skor (%):**

Ortalama F1-Skor değeri %48'dir. F1-Skor, precision ve recall değerlerinin harmonik ortalamasını temsil eder. Bu değer, modelin sınıfları hem doğru hem de kapsamlı bir şekilde tahmin etmede orta seviyede başarılı olduğunu gösterir.

Sonuç olarak, modelin genel performansı orta seviyede bir doğruluk, precision, recall ve F1-Skor ile karakterize edilmektedir. Ancak, iyileştirme adımları, belirli sınıfların performansını artırmak ve modelin genel etkinliğini optimize etmek amacıyla gerçekleştirilebilir.

## 5.3. Random Forest Algoritması Sonuçları

TAHMİNİ SINIF

GERÇEK

SINIF

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Sınıf 1 | Sınıf 2 | Sınıf 3 | Sınıf 4 | Sınıf 5 |
| Sınıf 1 | 293 | 91 | 26 | 13 | 2 |
| Sınıf 2 | 104 | 185 | 88 | 41 | 6 |
| Sınıf 3 | 43 | 93 | 167 | 92 | 24 |
| Sınıf 4 | 15 | 31 | 77 | 191 | 93 |
| Sınıf 5 | 11 | 22 | 24 | 86 | 275 |

**Tablo 5.2.1** Random Forest algoritmasının hata matrisi

**Genel Model Performansı:**

Toplam 2093 test örneği üzerinde modelin accuracy (doğruluk) değeri %53 olarak elde edilmiştir. Bu, toplamda doğru sınıflandırılan örneklerin oranını gösterir.

**Sınıf Bazında Analiz:**

Sınıfların precision, recall ve f1-score değerleri incelenmiştir. Örneğin, sınıf 1 için precision %63, recall %69 ve f1-score 0,66 olarak bulunmuştur. Bu değerler, sınıf 1'in model tarafından ne kadar iyi tahmin edildiğini gösterir.

**Dengesiz Sınıflar:**

Sınıflar arasında dengesizlik gözlemlenmiştir. Özellikle, sınıf 5'in recall değeri diğer sınıflara göre daha yüksektir, bu da modelin sınıf 5'i daha iyi tanıdığını gösterir.

**Düşük Performanslı Sınıflar:**

Sınıf 2, sınıf 3 gibi düşük performanslı sınıfların düşük precision, recall ve f1-score değerleri vardır. Bu sınıfların model tarafından zayıf bir şekilde tahmin edildiği anlaşılmaktadır.

**Genel Değerlendirme:**

Weighted avg (ağırlıklı ortalama) değerlendirmesine göre, modelin genel performansının orta seviyede olduğu görülmektedir. Ancak, iyileştirme alanları bulunmaktadır, özellikle düşük performanslı sınıflarda performansı artırmak önemlidir.

**İyileştirme Alanları:**

Modelin performansını artırmak için özellikle düşük performanslı sınıflarda iyileştirmeler yapılması gerekmektedir. Bu, belirli sınıfların daha fazla örnek ile eğitilmesi, model parametrelerinin ayarlanması veya farklı bir modelin denemesi gibi stratejileri içerebilir.

|  |  |
| --- | --- |
| Performans Sonuçları | Yüzde (%) |
| Ortalama Accuary | 53 |
| Ortalama Precesion | 53 |
| Ortalama Recall | 53 |
| Ortalama F1-Skor | 53 |

**Tablo 5.3.2** Random Forest algoritmasının Performans Sonuçları

**Ortalama Accuracy (%):**

Modelin, 5 farklı sınıfı doğru bir şekilde sınıflandırma oranı %53'tür. Bu değer, toplam örneklerin yarısından biraz daha fazlasının doğru bir şekilde tahmin edildiğini göstermektedir.

**Ortalama Precision (%):**

Ortalama precision değeri %53'tür. Bu, modelin sınıfları doğru bir şekilde tahmin etme yeteneğinin orta seviyede olduğunu gösterir. Precision, bir sınıfın tahminlerinin ne kadarının gerçekten doğru olduğunu ölçer.

**Ortalama Recall (%):**

Ortalama recall değeri %53'tür. Bu, modelin gerçek pozitif örneklerin yarısından biraz daha fazlasını yakalayabildiğini gösterir. Düşük bir recall değeri, özellikle sınıfların tanınma oranının orta seviyede olduğunu gösterir.

**Ortalama F1-Skor (%):**

Ortalama F1-Skor değeri %53'tür. F1-Skor, precision ve recall değerlerinin harmonik ortalamasını temsil eder. Bu değer, modelin sınıfları hem doğru hem de kapsamlı bir şekilde tahmin etmede orta seviyede başarılı olduğunu gösterir.

Sonuç olarak, modelin genel performansı orta seviyede bir doğruluk, precision, recall ve F1-Skor ile karakterize edilmektedir. Ancak, iyileştirme adımları, belirli sınıfların performansını artırmak ve modelin genel etkinliğini optimize etmek amacıyla gerçekleştirilebilir.

**6. SONUÇLAR**

Yapılan testler sonucunda üç farklı sınıflandırma algoritması (KNN, Decision Tree, Random Forest) kullanılmıştır. Genel olarak:

* KNN Algoritması: %44 doğruluk oranı elde etmiştir. Sınıflar arasında dengesizlik gözlemlenmiş, özellikle sınıf 3 ve sınıf 4 düşük performans göstermiştir.
* Decision Tree Algoritması: %48 doğruluk değeri ile genel performans orta seviyede bulunmuştur. Sınıf 1 için iyi bir tahmin performansı gösterirken, düşük performanslı sınıfların sayısında azalma sağlanmıştır.
* Random Forest Algoritması: %53 doğruluk oranı ile en yüksek performansı sergilemiştir. Sınıf 1 için yüksek precision ve recall değerleri elde edilmiş, genel olarak daha dengeli bir tahmin performansı sergilenmiştir.

**Doğruluk (Accuracy):**

* KNN: %44
* Decision Tree: %48
* Random Forest: %53

Doğruluk oranlarına bakıldığında, Random Forest algoritmasının diğerlerine kıyasla daha yüksek bir doğruluk elde ettiği görülmektedir.

**Kesinlik (Precision):**

Her üç algoritmanın sınıflar bazında kesinlik değerleri incelendiğinde, sınıflar arasında farklı performanslar elde edilmiştir. Özellikle Decision Tree ve Random Forest modelleri, belirli sınıflarda daha yüksek kesinlik değerlerine sahiptir.

**Duyarlılık (Recall):**

* KNN: Düşük performanslı sınıflarda düşük recall değerleri
* Decision Tree: Genelde dengeli bir recall performansı, sınıf 5'te daha yüksek
* Random Forest: Sınıf 5'te yüksek recall değeri, diğer sınıflarda da dengeli bir performans

**F1-Skor (F1-Measure):**

* KNN: Düşük performanslı sınıflarda düşük F1-Skor değerleri
* Decision Tree: Ortalama bir F1-Skor, sınıf 5'te daha yüksek
* Random Forest: Sınıf 1 ve 5'te yüksek F1-Skor değerleri, genel olarak daha dengeli bir performans

**Genel Değerlendirme:**

* Her üç algoritma da belirli sınıflarda düşük performans göstermiştir.
* Random Forest, diğer algoritmalara göre genel doğruluk, kesinlik, recall ve F1-Skor açılarından daha iyi bir performans sergilemiştir.
* Sınıflandırma performansının iyileştirilmesi amacıyla, özellikle düşük performanslı sınıfların üzerinde durularak modelin geliştirilmesi gerekmektedir.
* Modellerin performansı, kullanılan algoritmanın yanı sıra veri setinin özelliklerine ve sınıf dengesizliklerine bağlı olarak değişkenlik göstermektedir.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Ortalama Accuracy (%)** | **Ortalama**  **Precision (%)** | **Ortalama**  **Recall (%)** | **Ortalama**  **F1-Skor (%)** |
| KNN | 44 | 45 | 44 | 44 | |
| Desicion Tree | 48 | 48 | 48 | 48 | |
| Random Forest | 53 | 53 | 53 | 53 | |

**Tablo 6.1.** Kullanılan tüm algoritmalar için ortalama performans sonuçları

# Bu veri kümesi için, incelenen metriklerde en yüksek ve en düşük değerlere odaklanarak şu değerlendirmeleri yapabiliriz:

# Bu veri kümesi için Random Forest modeli genel performans açısından en iyi sonuçları vermektedir. En düşük değerlere bakıldığında ise tüm metriklerde en düşük değerlere sahip olan model KNN'dir (%44).

# 7. ÖNERİLER

Bu çalışmamızda verileri çekerken sadece İstanbul şehrinin belirli semtlerindeki verileri çektik bu yüzden verilerimizin sayısı kısıtlıydı. Verileri çekerken tüm Türkiye’deki şehirleri semtleri ve mahalleri çekerek daha büyük bir veri seti elde ederek algoritmanın daha iyi eğitilmesi sağlanabilir.

Verileri çekerken evlerin belirli özelliklerini çektik. Daha çok ev özelliği çekerek eğitilen modelin başarı oranı daha da arttırılabilir.

Eğitim için kullanılan algoritmanın parametrelerini değiştirerek veya farklı daha karmaşık verileri çözen bir algoritma kullanarak başarı oranı arttırılabilir.

# 8. KAYNAKLAR

[1] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., 2023, Scikit-learn: Machine Learning in Python[online], 2.0.0 , Journal of Machine Learning Research , <https://scikit-learn.org/stable/> , [Ziyaret Tarihi: 28 Aralık 2023].

[2] McKinney, W. (2017). Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython (2.a ed.). O'Reilly Media., <https://pandas.pydata.org/pandas-docs/stable/> [Ziyaret Tarihi: 28 Aralık 2023].

[3] Hunter, J. D. (2022). Matplotlib: A Comprehensive Guide to Creating Figures with Python (4.a ed.). Packt Publishing., <https://matplotlib.org/stable/> [Ziyaret Tarihi: 28 Aralık 2023].

[4] Yazar: Erdinç, S. (2023). Python BeautifulSoup Modülü [online]., YazBel. ,<https://www.sinanerdinc.com/python-beautifulsoup-> [Ziyaret Tarihi: 28 Aralık 2023].

[5] Reitz, M. (2021). Python Requests: HTTP for Humans [online]. O'Reilly, https://requests.readthedocs.io/en/master/ [Ziyaret Tarihi: 2024-01-05].

[6] Müller, A. C., & Guido, S. (2020). Python Data Science Handbook: Essential Tools for Working with Data [online]. (Edition), O'Reilly., [Ziyaret Tarihi: 30 Aralık 2023].,

https://jakevdp.github.io/PythonDataScienceHandbook/: [Ziyaret Tarihi: 2024-01-05].

[7] Ulgen E.K (2017), Makine Öğrenimi Bölüm-5 (Karar Ağaçları), medium, <https://medium.com/@k.ulgen90/makine-%C3%B6%C4%9Frenimi-b%C3%B6l%C3%BCm-5-karar-a%C4%9Fa%C3%A7lar%C4%B1-c90bd7593010> [Ziyaret Tarihi:30 Aralık 2023].

[8] Öztürk M., Rastgele Orman (Random Forest), <https://miracozturk.com/python-ile-siniflandirma-analizleri-rastgele-orman-random-forest-algoritmasi/> , [Ziyaret Tarihi:31 Aralık 2023].

[9] Durna M.B (2020), Cross-Validation nedir? Nasıl çalışır? Medium , <https://medium.com/bili%C5%9Fim-hareketi/cross-validation-nedir-nas%C4%B1l-%C3%A7al%C4%B1%C5%9F%C4%B1r-4ec4736e5142> [Ziyaret Tarihi:31 Aralık 2023].

[10] <https://www.emlakjet.com/satilik-konut/istanbul/>