Prof.Dr. Mehmet Fatih AMASYALI BLM3760 Uzman Sistemlere Giriş

**Son Teslim tarihi:** **05.01.2023 23:59**

Uzman Sistemlere Giriş Projesi

Oğuzhan Topaloğlu

Ç19052025 – Grup 1

*Bilgisayar Mühendisliği Bölümü,*

*Elektrik-Elektronik Fakültesi,*

*Yıldız Teknik Üniversitesi*

****

Istanbul, 2023

|  |
| --- |
| **İÇERİK** |
| 1. Genel Bilgiler |
| 2. Geliştirme Sürecinde Yaşananlar |
| 3. Menü Fonksiyonlarının Çalıştırılması |
| 4. Sistemin Sayısal Başarısı |
| 5. Kaynaklar |
| 6. Youtube Videosu |
| 7. Program Dosyalarının Listesi |
|  |
|  |

**1. Genel Bilgiler**

Bu projede vatan bilgisayardan telefonlarla ilgili bilgileri web scraping yaparak okuyan, Random Forest tekniğini ve bu verileri kullanarak bir model oluşturan ve bu modeli kullanarak özellikleri verilen bir telefonun fiyatını tahmin eden bir uzman sistem geliştirilmiştir.

Projede görsel arayüz yerine terminal ve dil olarak Python kullanılmıştır. Kütüphane olarak veritabanı oluşturmada pandas, model oluşturmada sklearn, web scraping’de de beautifulsoup4 (bs4) ve requests (HTTP kütüphanesi) kullanılmıştır.

Proje 4 adet dosyadan oluşmaktadır: main.py, menu.py, scrape.py, expert.py.

main.py dosyası, içinde terminalde yazı yazdıran print işlemlerini ve girdilere göre yapılacak işlemleri içerir. Menüleri yazdırırken main dosyasının çok kalabalık olmamasını sağlama amacıyla veri tutarlılığı kontrollerini yapan kodlar menu.py dosyasına konmuştur.

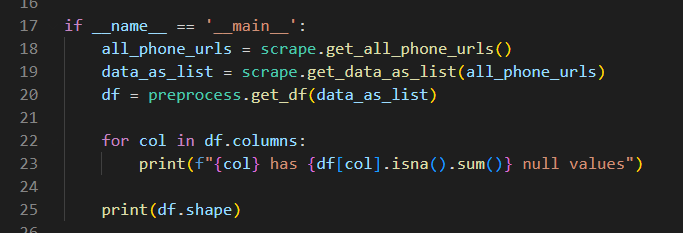
scrape.py dosyası, Vatan Bilgisayar’dan telefonlarla ilgili verileri çekip list[dict[str:str]] olarak dönen fonksiyonları içerir.

expert.py dosyası da içinde scrape.py’da elde edilen listeleri alıp veritabanı oluşturup, preprocessing yapıp (gelen datanın gereksiz kısımlarını atma, veri türlerini tutarlı yapma vb.), model oluşturup, bu model ile fiyat tahmini yapan fonksiyonları barındırır.

**2. Geliştirme Sürecinde Yaşananlar**

Proje sırasında en çok preprocessing kısmında zorlandım ve uzman sistemlerin geliştirilmesi sürecinde bilginin depolanmasının (bilgi mühendisi olmanın) ne kadar zor bir şey olduğunu fark ettim. Elde ettiğimiz veriler genelde yanlış veri türlerinde olabiliyor ve bu da RandomForestClassifier gibi sınıflarda hata almamıza neden oluyor. Ayrıca çektiğimiz verilerde bir sürü null oluşabiliyor ve bu da modelin düzgün çalışmamasına neden oluyor.

Projenin başlarında Vatan Bilgisayar’dan her veriyi çekiyor ve veritabanıma koyuyordum. Ancak sonradan fark ettim ki, bu çektiğim verilerin çoğunluğu null’lardan oluşuyor. Bunun nedeni her telefon sayfasında farklı bilgilerin verilebiliyor olmasıydı. Bazıları ayrıntılı bir şekilde bilgi tablolarını dolduruyorken bazıları yalnızca isim, fiyat, boyut, ağırlık, menşei gibi bilgileri içeriyordu. Ben de bu null durumunun önüne geçmek için en az null içeren sütunları yazdıracak şu kodu yazdım:

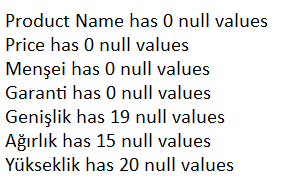


Bu kod ile aldığım çıktıları da terminalden kopyala yapıştır ile bir txt’ye koydum ve az satır var diye manüel olarak null sayılarını kontrol ettim.

Bu txt dosyası şöyle gözüküyordu:

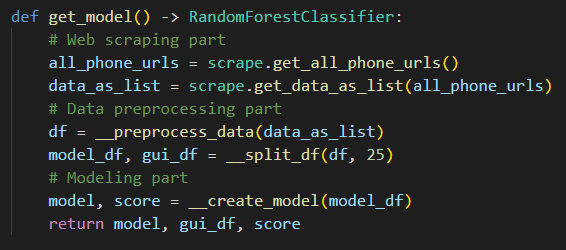


Gittim ve dediğim gibi manüel olarak aşağıdaki görseldeki az null olan sütunları belirledim:



Buradan da 5 tane ek bilgimin olacağı ortaya çıkmış oldu, bunlar menşei, garanti, genişlik, ağırlık ve yükseklik idi.

Ardından verileri okudum, preprocessing işlemlerini yaptım ve veritabanını iki parçaya ayırdım:

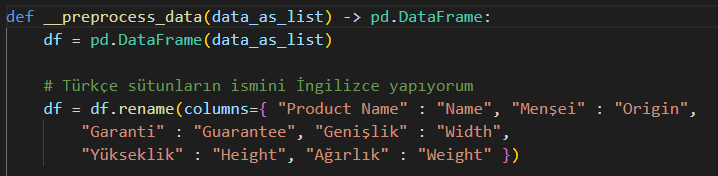


Konu belirleme mailinde bana şunu demiştiniz:

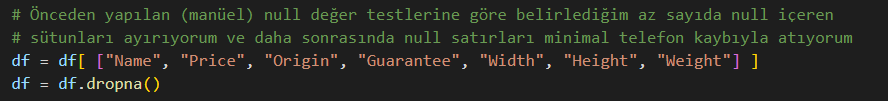


Bu yüzden veritabanını iki parçaya ayırırken bir veritabanına 25 (en az 20 dediniz, ben 25 seçtim), öbür veritabanına da geriye kalan satırları koydum.

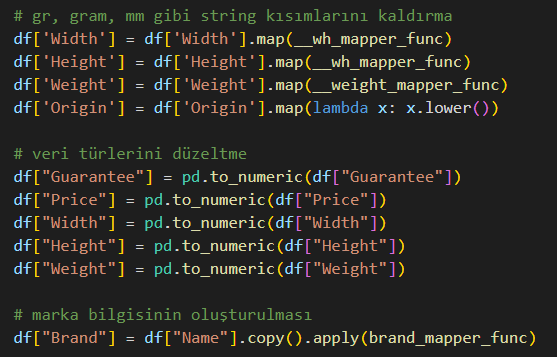
Burada model oluşturma kısmını anlatmadan önce preprocessing’de yaptıklarıma da kısaca değinmek istiyorum. expert.\_\_preprocess\_data() fonksiyonunda öncelikle listemi dataframe’e dönüştürüyorum ve Türkçe sütun isimlerini İngilizce’ye çeviriyorum:



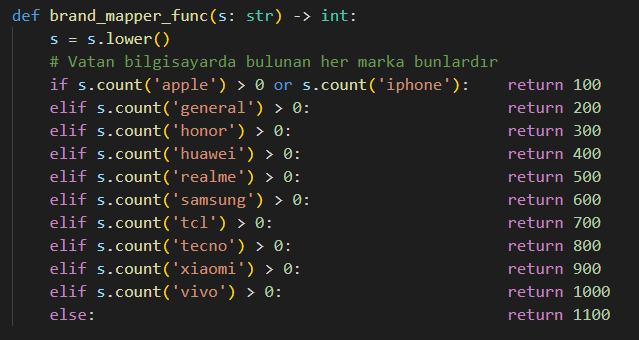
Daha sonra, önceden belirlediğim sütunlar dışındaki sütunları siliyorum ve en az bir null değer içeren her satırı siliyorum:

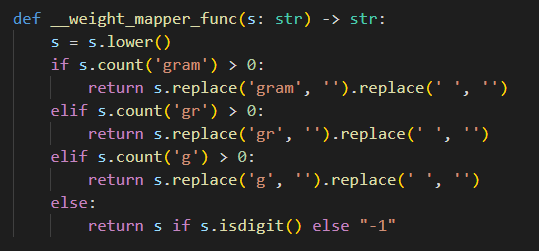


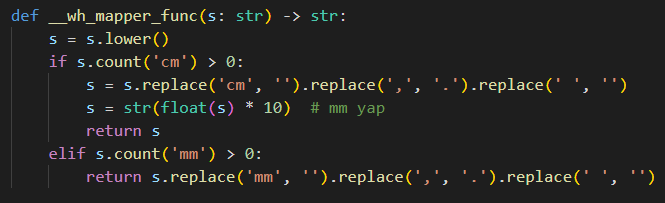
Bunları yaptıktan sonra fark ettim ki width, height ve weight girişi yapılırken de bir sürü farklı değerler girilmiş. Örneğin width için bazıları mm cinsinden bazıları cm cinsinden değer girmiş. Kimisi “gr” kimisi “Gr” kimisi de “gram” filan yazmış. Aynı durum height ve weight için de geçerliydi. Burada bu değerleri düzeltip, ortak bir ölçü birimine (mm ve gram seçtim) dönüştürüp sütunların veri türünü düzeltmem gerekiyordu:



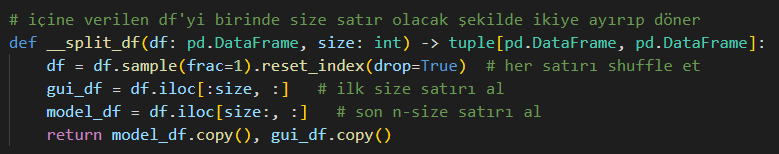
Bunlar için fonksiyonlar oluşturdum, ardından veri türünü nümerik yaptım ve en son olarak da kendim bir tane brand feature’ı ekledim.



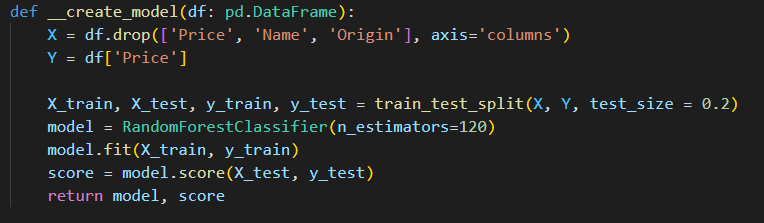




Ve ardından da dediğim gibi veritabanını ikiye ayırdım:



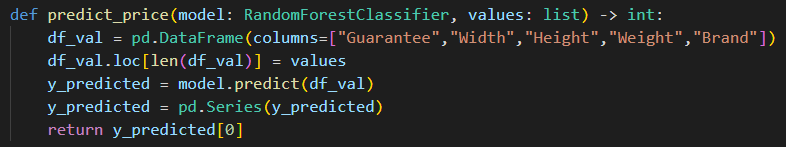
Ve sonra da model\_df kısmını kullanarak modelimi Random Forest ile oluşturdum:



Burada modelin, test setinde (25 telefon değil, model\_df’ten ayrılan test setinde) ne kadar başarılı olduğunu da test ettim ve score olarak döndüm.

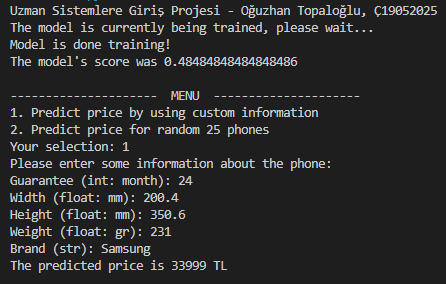
Burada kısaca Random Forest tekniğini de açıklamak istiyorum. Ben bu tekniği, bu dönem aldığım Veri Madenciliğine Giriş dersinde gördüm. Kısaca açıklamak gerekirse, Random Forest, bir karar ağacı yöntemine benzer bir model oluşturma yöntemidir. Bu yöntem birden fazla karar ağacını bir araya getirerek, ağaçlar arasındaki ortak karar verme süreciyle model oluşturmayı amaçlar. Her bir karar ağacı, veri setinden rastgele seçilen örnekler üzerinden eğitilir ve her bir ağaç, veri setindeki tüm örnekler için tahminler üretir. Tahminler, birleştirilerek, ortak karar verme süreciyle bir sonuç üretilir. Bu yöntem, özellikle büyük ve karmaşık veri setlerinde oldukça etkilidir ve yüksek doğruluk oranları elde edilebilir.

Bütün bunları yaptıktan sonra da main.py menüsünde çağrılacak fiyat tahmin eden fonksiyonumu tanımladım:



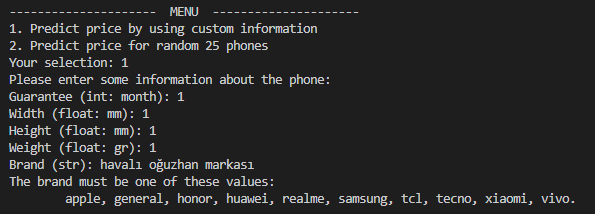
**3. Menü Fonksiyonlarının Çalıştırılması**

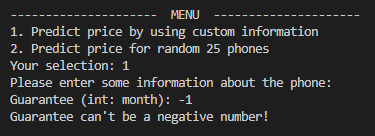
Menü aşağıdaki görseldeki gibi görünmektedir:

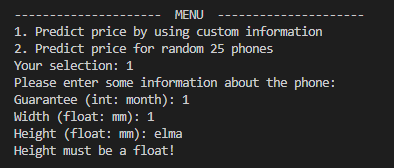


Bu görselde 1 seçeneğini seçtim ve bir tahmin yaptırdım. Bu seçenek bizden garanti, genişlik, yükseklik, ağırlık ve marka bilgilerini girmemizi istiyor ve bir fiyat tahmini yapıyor. Burada arkaplanda bir sürü veri tutarlığı kontrolü de yapıyorum. Girilen değerler gerçekten de int, float, str şeklinde olmalı ve negatif gibi değerler bulunmamalıdır. Brand kısmında olmayan bir marka girilirse de girilebilecek marka türleri yazdırılıyor ve kullanıcıya bilgi veriliyor. (NOT: bu marka listesini doğrudan Vatan Bilgisayar’dan aldım ve bu liste asla değişmiyor, gelecekte değişirse diye nolur nolmaz diye “other” için bir sayı da tanımladım, lütfen bu ayrıntılar için koda bakınız)

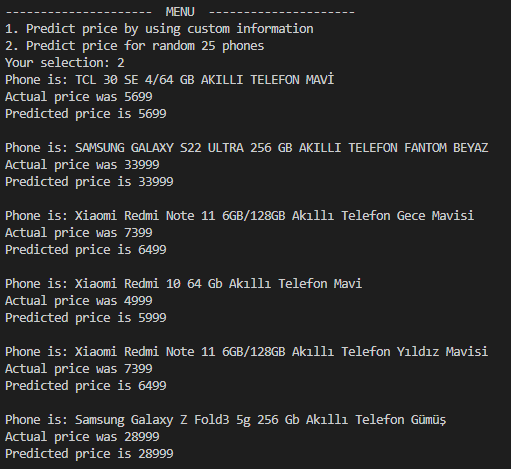
Yanlış veri girişi örnekleri:





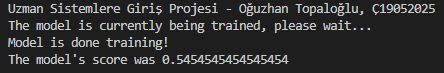


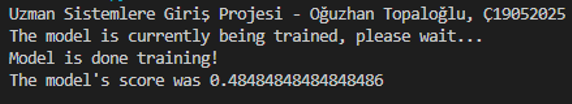
Burada ikinci seçenek seçildiğinde de otomatik olarak **rastgele** ayrılan 25 tane telefon için fiyat kontrolü yapılır:



**4. Sistemin Sayısal Başarısı**

Oluşturulan modelin skoru Vatan Bilgisayar’da az telefon (yaklaşık 150 adet) bulunmasından dolayı çok yüksek olmamaktadır. Genelde %45-%70 arasında bir skora sahip olduğunu kodu çalıştırarak gözlemledim:





Burada ayrıca ayrılan 25 telefonla ilgili başarı ölçümü yaparken de confusion matrix ve classification report (precision, recall, accuracy ve f1 score yazdırımı yapıyor) kullandım.

Confusion Matrix, bir sınıflandırma modelinin performansını değerlendirmede kullanılan bir ölçümdür. Bu matris, bir sınıflandırma modelinin tahminlerini gerçek sınıflarla karşılaştırarak, modelin ne kadar doğru tahmin ettiğini gösterir. Confusion Matrix, 2 boyutlu bir tablo şeklinde gösterilir ve sıklıkla binary (ikili) sınıflandırma problemlerinde kullanılır. Ancak, birden fazla sınıf için de kullanılabilir. Bu projede, birden çok binary olmayan sınıf için kullanılmıştır.

Confusion Matrix'te, satırlar gerçek sınıfları temsil ederken, sütunlar ise tahmin edilen sınıfları temsil eder. Matrisin sol üst köşesi, "True Positive" (TP) olarak adlandırılan doğru pozitif tahminleri, sağ üst köşesi ise "False Positive" (FP) olarak adlandırılan yanlış pozitif tahminleri gösterir. Sol alt köşesi ise "True Negative" (TN) olarak adlandırılan doğru negatif tahminleri, sağ alt köşesi ise "False Negative" (FN) olarak adlandırılan yanlış negatif tahminleri gösterir.

Precision, bir sınıflandırma modelinin tahminlerinin doğruluk oranını ölçen bir metriktir. Precision, True Positive sayısını, True Positive ve False Positive sayılarının toplamına bölerek hesaplanır.

Matematiksel olarak, precision şu şekilde ifade edilebilir: Precision = TP / (TP + FP)

Recall, bir sınıflandırma modelinin gerçek pozitifleri ne kadar iyi tahmin ettiğini ölçen bir metriktir. Recall, True Positive sayısını, True Positive ve False Negative sayılarının toplamına bölerek hesaplanır.

Matematiksel olarak, recall şu şekilde ifade edilebilir: Recall = TP / (TP + FN)

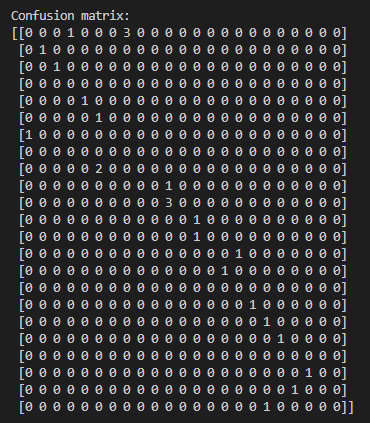
Accuracy, bir sınıflandırma modelinin tüm tahminlerinin doğruluk oranını ölçen bir metriktir. Accuracy, True Positive ve True Negative sayılarını, tüm tahminlerin (True Positive, True Negative, False Positive, False Negative) toplamına bölerek hesaplanır.

Matematiksel olarak, accuracy şu şekilde ifade edilebilir: Accuracy = (TP + TN) / (TP + TN + FP + FN)

F1 Score, precision ve recall değerlerinin harmonic ortalamasıdır ve sınıflandırma modelinin tahmin performansını ölçmek için kullanılır. F1 Score, precision ve recall değerlerine aynı önem verdiğinden, her iki değerin de yüksek olmasını sağlamaya çalışır.

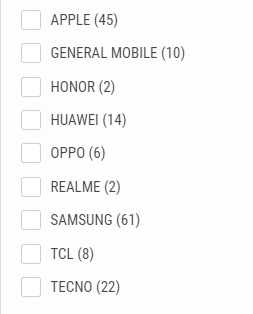
Matematiksel olarak, F1 Score şu şekilde ifade edilebilir: F1 Score = 2 \* (Precision \* Recall) / (Precision + Recall)

Sistemde ikinci seçenek seçildiğinde bu değerler yazdırılmaktadır ancak Vatan Bilgisayar’da bulunan telefonların sayısının az olması nedeniyle bir sürü değer 0 ve 1’e otomatik olarak atanmakta ve hesaplanamamaktadır. Bunun nedeni yukarıdaki matematiksel formüllerde 0’a bölünmeye çalışılmasıdır. Ayşe hocam Veri Madenciliği dersinde gerçek hayatta bazen bu 0’a bölmeden kaçınılması için 1 ekleniyor gibi bir şey demişti ancak ne kadar doğru bilmiyorum. Sklearn kütüphanesi böyle bir şey yapmıyor ve 0/1 değerlerini yapıştırıp bir uyarı mesajı print ediyor:

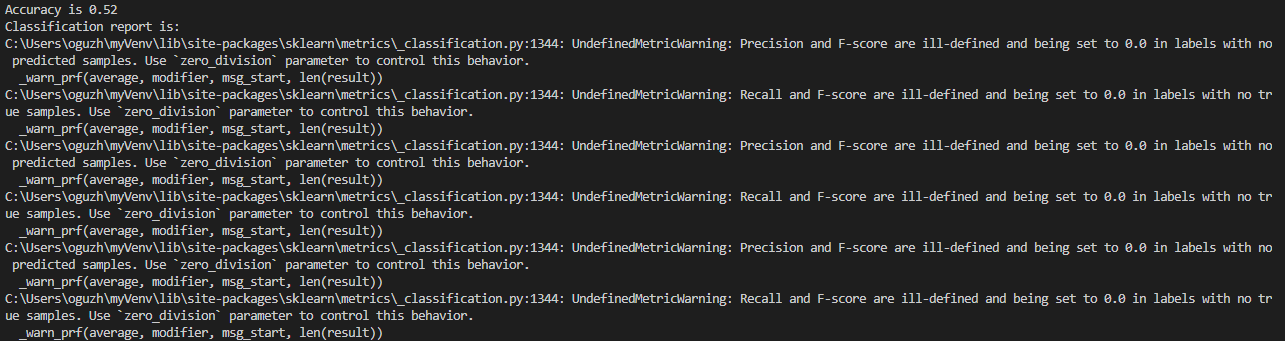


Dediğim gibi böyle sonuçların elde edilmesinin nedeni belirli markalardan çok telefon olması ve öbür markalardan çok telefon olmaması.

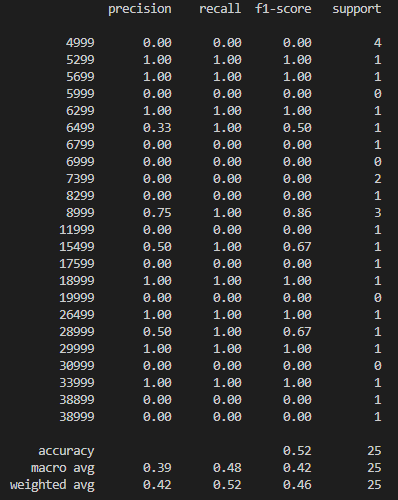
Vatan bilgisayarda Apple, Samsung, Xiaomi gibi markalardan çok telefon varken OPPO, General Mobile gibi markalardan telefon çok bulunmuyor:



Bu da metriklerin yazdırılırken aşağıdaki uyarıları basmasına neden oluyor:



Ancak bu uyarılara rağmen sklearn yine de metrikleri bize bir tablo ile veriyor:



**5. Kaynaklar**

Projede kaynak olarak sklearn dokümantasyonunu ve stackoverflow’u kullandım. Özellikle preprocessing sırasında kullandığım fonksiyonları stackoverflow’da öğrendim.

**6. Youtube Videosu**

Link: <https://www.youtube.com/watch?v=BTy5sTAC0vg> (5:31 süreli)

**7. Program Dosyalarının Listesi**

Projede şu dosyalar vardır: src klasörü (main.py, expert.py, scrape.py, menu.py) ve bu pdf.

NOT: projeyi çalıştırırken “python main.py” komutunu terminalde kullanabilirsiniz. Projede Python 3.9.0 kullandım ancak 3 sonrası her sürümün çalışacağını düşünüyorum. :)