Petrol Tüketiminin Makine Öğrenmesi İle Tahmin Edilmesi

line 1: 1st Ali Celi  
line 2: *dept. name of organization (of Affiliation)*  
line 3: *name of organization (of Affiliation)*line 4: City, Country  
line 5: email address or ORCID

Özet—Petrol tüketimi, ekonomik büyümenin ve aynı zamanda hava kirliliğinin göstergelerinden birini oluşturmaktadır. Petrolün sanayi üretiminde girdi ve enerji olarak kullanılması nedeniyle petrol tüketimi ve ekonomik büyüme arasındaki ilişki her dönemde araştırma konuşu olmuştur. Bu çalışmada, petrol verigisi, kişi başı gelir, asphalt yolların mil sayısı ve ehliyet sahiplerinin nüfusa oranına bakarak petrol tüketimini tahmin edeceğiz.

Abahtar Kelimeler—Petrol, Tüketim, Enerji, Tahmin

# GİRİŞ

Petrol tüketimi, insanlık tarihinde özellikle sanayileşme ve araçların icadından sonra önemli şekilde artmış ve sürekli talep edilen ürün haline gelmiştir. İnsanlar’da hayatlarını kolaylaştırmak için kullandığı araçlarda petrol kullanmış ve tüketim her geçen sene daha da artmıştır. Elbette böyle bir durumun ardından ülkeler düzenlemeler ve bu kullanılan petrolden vergiler almaya başlamışlar. İnsanların seyahat ettiği yollarda uzadıkça petrole olan talep artmıştır. Ayrıca petrol sadece insanların kullandığı araçlar için değil büyük fabrikaların da yakıt kaynağı olmuştur. Bu proje de petrol tüketiminin tahminini yapacağımız verisetinde insaların o bölgedeki özelliklerine göre petrol tüketim eğilimini hesaplayacağız ve bunları görselleştireceğiz.

Yapacağımız tahmin algoritmalarında kullanacağımız 5 adet regresyon modellerimiz Lineer Regresyon, K-Yakın Komşuluk Regresyon, Karar Ağacı Regresyon, Rassal Orman Regresyon ve Destek Vektör Makine Regresyonudur. Bu regresyon yöntemleriyle öncelikle verisetimizdeki modeli eğiteceğiz ve ardından test edip metrik sonuçlarına bakacağız.

Hedefimiz bu regresyon modelleriyle eğittiğimiz verisetimizin ortalama mutlak skorunu ve ortalama kare hatasının karekökünün düşük seviyelerde olması, model testlerin ardından en iyi modeli seçip bu model üzerinden farklı metriklerde incelenebilir.

Başarı Kriterimiz ise mean absolute error ve root mean squared error değerlerimiz olacağından bu değerlerde mean absolute error için 40-80 arası, root mean squared error metriği için ise verisetmize bağlı olarak 6-10 arasıdır.

# VERİ SETİ

## Veri Kaynağı

Veri setimiz açık kaynak kodlu bir verisetidir ve bu veri seti Kaggle platformundadır. <https://www.kaggle.com/datasets/harinir/petrol-consumption> adresinde bulunan verisetini projem için kullandım. Bu internet sitesinde bulunan veri seti CCO: Public Domain lisanslı ve herkes indirebilir.

## Veri Kümesi

* Veri kümemiz 5 sütun ve 48 satırdan oluşmaktadır.
* Veri kümesindeki sütunlar sırasıyla; Petrol\_tax(Petrol’den alınan vergilerin değerleri), Average\_income(Kişinin aldığı ortalama geliri), Paved\_Highways(Bölgenin otoyol uzunluğu), Population\_Driver\_license(%)(Popülasyonun ortalama ehliyetli kişi sayısına oranı) ve Petrol\_Consumptiondır(Petrol tüketimini temsil ediyor).
* Veri kümesinde herhangi bir boş yada aykırı bir veri yoktur.
* Veri kümesinin dosya boyutu 1.21 kilobayt’dır.

## Öznitelik Açıklamaları

Veri kümemizde herhangi bir hatalı veya eksik verimiz yok eğer olsaydı bunları doldurabilirdik veya verisetinin durumuna göre bunları verisetimizden kaldırabilirdik. Sütunların yapısı olarak tüm sütunlarımız sayısal veri tipindedir bundan dolayı herhangi bir label encoding işlemi kullanmamıza gerek yok. Petrol\_tax ve Population\_Driver\_license(%) sütunlarımız float(ondalıklı sayı) tipinde, diğer 3 sütunumuz(Average\_income,Paved\_Highways ve Petrol\_Consumption) ise int(tam sayı) veri tipindedir.

## Sıralama, Kategorizasyon

Veri kümemizdeki veriler herhangi bir sırada veya kategoride değiller, average\_income’lara bakarak o kişilerin gelir durumuna göre sıralayabilir ve petrol tüketiminin korelasyonuna bakabiliriz. Aynı şekilde Petrol\_tax değerine bakarak o bölgedeki vergi oranına göre de petrol tüketimleri incelenebilir.

## Öznitelikler Arasındaki Korelasyonlar

Graphical user interface, application

Description automatically generated

Öznitelikler arasındaki korelasyonlara bakıldığında pozitif korelasyon olarak en yüksek değer Population\_Driver\_license(%) ile Petrol\_Consumption arasında olmuş, bu da bize söylemek istiyor ki bir bölgede kişi başına düşen ehliyet sayısı arttıkça petrol tüketimi de o bölgede artış sağlıyor. Negatif korelasyon tarafına bakarsak burada da en yüksek değer Petrol\_tax ile Paved\_Highways arasında olmuş buradan da petrol vergisinin düşük olduğu yerlerde asfalt yolların daha çok olduğunu çıkarabiliriz.

# KULLANILAN YÖNTEMLER

Projemizde kullanılan yöntemler ilk olarak verisetimizi Jupyter Notebook idesinde açmak. Projemizi Jupyter ortamında açmak için öncelikle Jupyter ortamına kütüphanelerimizi import ediyoruz.

Graphical user interface, text

Description automatically generated with medium confidence

Buradaki kütüphanelerden bahsetmek istiyorum, lineer işlemler ve diğer matematiksel denklemleri kullanmak için numpy kütüphanesini, veri kümemizi tanımlamak ve üzerinde işlem yapmak için pandas’ı, görselleştirme ve model sonuçlarının analizlerini görmek için matplotlib kütüphanesinin içinde bulunan pyplot’u, veri kümesindeki eğitim değerlerimi ölçekleyebilmek için scikit-learn içinde bulunan StandartScaler fonksiyonunu, yine veri kümemizde ki verileri eğitim ve test olarak bölmek için train\_test\_split fonksiyonunu, regresyon modellerimden olan Lineer Regresyon için LinearRegression fonksiyonunu, K-En Yakın Komşuluk için KneighborsRegressor fonksiyonunu, Karar Ağaçları modeli için DecisionTreeRegressor fonksiyonunu, Rassal Ağaçlar modeli için RandomForestRegressor fonksiyonunu, Destek Vektör Makineleri için SVR fonksiyonunu kütüphaneme import ediyorum. Bunların haricinde eğiteceğim modellerin metrik sonuçlarını görmek için mean\_absolute\_error ve mean\_sqaured\_error fonksiyonlarını projeme dahil etmek için import ediyorum.

A picture containing table

Description automatically generated

Bir sonraki adımda ise verisetimi Jupyter iş istasyonuna yüklüyorum ardından dataset değişkenine atama yapıyorum, dataset.head() fonksiyonuyla verisetimdeki ilk 5 satır değeri ekrana basıyorum.

Table

Description automatically generated

Daha sonra veri kümemin keşifsel analizini yapmak için describe fonksiyonunu kullanıyorum. Bu fonksiyon bana verilerimin sayısını, ortalamasını, standart sapmasını, minimum ve maksimum değerleri gibi sonuçları sütun bazlı olarak gösterecek.

Graphical user interface

Description automatically generated with low confidence

Veri kümemdeki boş değerlerinin kontrolünü yapmam için dataset.isnull() fonksiyonunu kullanmam gerekiyor, bu bana her satırda boş var mı yok mu bunun kontrolünü yapacak eğer boş değer varsa True yoksa False değerini gönderecek bu fonksyionun en sonuna sum() fonksiyonunu da eklersem her sütundaki toplam boş değeri bana gösterecek. Görüldüğü üzere veri kümemde herhangi bir boş değerim yok.

A picture containing graphical user interface

Description automatically generated

Bu işleminde ardından veri setimin önişleme adımları bitmiş oluyor ve modele bu veri kümemi hazırlamak için hedef değerimi ve özellik değerimi ayırıyorum burada petrol tüketim tahmini yapacağımız için Petrol\_Consumption sütunu hedef değerimiz ve geri kalan 4 sütun ise özellik sütunumuz olmalı, X değerime bu 4 sütunu y değerime ise Petrol\_Consumption değerlerimi atıyorum.

Ardından X de bulunan ilk 4 sütun değerlerimin hepsini ölçeklendirmem gerekiyor çünkü küçük sayılarla çalışmak model eğitiminde hız ve doğruluk açısından çok önemli, bundan dolayı import ettiğim StandartScaler() fonksiyonumu çağırıp sc değerime atıyorum ardından bu değerimin içinde bulunan fit\_transform ile X değerimi ölçeklendiriyorum.

Bu adımın ardından KFold fonksiyonunu çağırarak veri kümemi 10 parçaya bölecek fonksiyonu yazıp ardından cv değerime atıyorum.

Graphical user interface, text, application

Description automatically generated

Model eğitimine geçebiliriz, ilk olarak Lineer Regresyon fonksiyonum için scores\_lr adında bir değişken kullanıyorum ve bu değişkene modelimi inşa ediyorum. Scoring değeri olarak neg\_mean\_absolute\_error değerimi seçiyorum, ve daha sonra bu model değerimin sonucunu önce np.absolute fonksiyonuna koyarak mutlak değerini ardından np.mean ile ortalama sonucunu alıyorum ve bu da aslında mean\_absolute\_error dediğimiz değeri bize sağlıyor. Aynı şekilde bu değeri alıp üstüne np.sqrt fonksiyonunu ekledikten sonra rmse metrik sonucumu elde ediyorum.

Graphical user interface, text, application

Description automatically generated

İkinci modelimiz K-Yakın Komşuluk Resgresyonu, bu regresyon algoritmanın çalışma yapısı öncelikle verilen değerleri bir kordinat uzayına nokta olarak yerleştiriyor ve bu değerlerin bağlı olduğu hedeflere bakarak en yakın komşularının öklid, manhattan yada minkowski uzaklık formüllerine göre hesaplayıp o değeri buluyor. Bizim algoritmamızda ise eğittiğimiz modeli 83.92 Mean Absolute Error ve 9.16 Root Mean Squared Error ile bu değer, bulmuş oluyoruz.

Graphical user interface, text, application

Description automatically generated

Karar Ağacı Regresyon Algoritmasının çalışma yapısı ise, verilen kümedeki değerleri bir ikili ağaca göre çizip o özellikleri eğitmek, yani aslında insanların kilosuna göre 90 kilo altı mı üstü mü, cinsiyeti kadın mı erkek mi gibi ikiye bölme yaparak bir karar ağacımız oluşuyor ve test aşamasında bu ikili ağaçlara bakarak tahminler yapıyoruz. Bu algoritmamız 74.83 Mean Absolute Error ve 8.65 Root Mean Squared Error değerlerine ulaşmış.

Graphical user interface, text, application

Description automatically generated

Rassal Ağaç Regresyonu da Karar Ağaçlarına benzer olarak çalışır ve aynı ikili ağaçlar üzerinden eğitim ve test işlemlerini yapar fakat bu algoritma diğerinden biraz daha karmaşıktır ve kolayca dallanıp budaklanabilir bu da bizim istemediğimiz şekilde overfitting sorununa yol açabilir. Bu algoritma bize 44.86 Mean Absolute Error ve 6.69 Root Mean Squared Error gibi diğerlerinden daha güzel metrik sonuçları sağlamıştır.

Graphical user interface, text, application

Description automatically generated

Destek Vektör Makineleri, bir margin değeri üzerinde o noktaların en uzağında bulunan değere göre tahmin yapan bir sistemi ortaya koyuyor. Burada Destek Vektör Makinelerinin metrik sonuçlarına baktığımızda, 80.26 Mean Absolute Error ve 8.95 Root Mean Squared Error gibi değerleri tahminlediğini ve hesapladığını görüyoruz.

# SONUÇLAR

Modelleri eğitmenin ardından her biri için tahminler yapılır ve bu tahminlerin ne kadar başarılı olduğunu öğrenmek için metrik sonuçları kullanılır. Metrik sonuçları olarak Mean Absolute Error(Ortalama Mutlak Hatası) ve Root Mean Squared Error(Ortalama Kareler Hatasının Karekökü) değerleri kullanılır.

* Mean Absolute Error: Bu metrik değeri modelin performansını ölçer ve her zaman pozitif değerlidir, 0 ile ∞ arasında değerler alır. Mean Absolute Error değeri genel olarak regresyon eğrisinin bir dizi noktaya ne kadar yakın olduğunu söyler.(Şekil 2.1)

Chart, bar chart

Description automatically generated

Şekil 2.1: Algoritmaların MAE Skorları

Root Mean Squared Error: Ortalama Kareler Hatasının Karekökü değeri tahminleyicinin, tahmin ettiği değerler ile gerçek değerler arasındaki uzaklığın bulunmasında sıklıkla kullanılan ve hatanın büyüklüğünü ölçen kuadratik bir metriktir. Ortalama Kareler Hatasının Karekökü için tahmin hatalarının standart sapması da diyebiliriz.(Şekil 2.2)

Chart, bar chart

Description automatically generated

Şekil 2.2: Algoritmaların RMSE Skorları

# TARTIŞMA, SONUÇ VE ÖNERİLER

Petrol tüketimi tüm dünyada her geçen gün değeri artıyor ve bundan dolayı çok fazla araştırma ve proje üzerinde çalışılıyor . Bu bağlamda, benim yaptığım projede, makine öğrenmesi algoritmaları kullanılarak verilen kullanıcı bilgilerinin analizi ve eğitimi yapılmıştır ardından tahmin algoritmaları çalıştırılmıştır. Linear Regression kullanılarak, 52.58 Mean Absolute Error değeri ve 7.25 Root Mean Squared Error Değeri, K-Nearest Neighbors kullanılarak 83.9 (MAE)Mean Absolute Error değeri, 9.16 RMSE(Root Mean Squared Error) değeri elde edilmiştir, bir diğer algoritma olan Decision Tree Regressor’da 74.83 MAE ve 8.65 RMSE değeri, Random Forest algoritmasında 44.8 MAE ve 6.69 RMSE değeri, Support Vector Machine algoritmasında 80.2 MAE ve 8.95 RMSE, Bu sonuçlara bakarak Random Forest algoritması en başarılı çalışan algoritma olmuştur diyebiliriz.

Bu çalışmanın ardından daha yüksek başarı oranlarına ulaşmak için , daha farklı optimizasyon yöntemleri, aktivasyon fonksiyonları, veri artırma(data augmentation) yada çeşitli diğer veri önişleme yapıları kullanarak modeli daha yüksek başarı oranları ile eğitip test edebiliriz. Yakıt sektöründe bir proje olduğu için bu projeyi konunun uzmanları ile konuşup fikir alışverişi yaparak üstüne farklı bakış açıları da eklenebilir. Aynı zamanda açık kaynak kodlu olarak belirli platformlarda projeyi paylaşarak bu konu üzerinde çalışan kişilere yardımcı olabiliriz.

##### KAYNAKLAR

1. Petrol Consumption Kaggle Veri Kümesi, <https://www.kaggle.com/datasets/harinir/petrol-consumption>
2. Standart Scaler Anlatımı, <http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html#:~:text=Standardize%20features%20by%20removing%20the%20mean%20and%20scaling%20to%20unit%20variance.&text=where%20u%20is%20the%20mean,or%20one%20if%20with_std%3DFalse%20>.
3. Makine Öğrenmesi Algoritmaları, <https://www.w3schools.com/python/python_ml_linear_regression.asp#:~:text=The%20term%20regression%20is%20used,the%20outcome%20of%20future%20events>.
4. Bar Plot Anlatımı, <https://www.geeksforgeeks.org/bar-plot-in-matplotlib/>