Makine Öğrenmesi Algoritmaları ile İkinci El Araç Fiyat Tahmini

Used Car Price Prediction with Machine Learning Algorithms

Samed ZIRHLIOĞLU
Bilgisayar Mühendisliği
18110131037
Kahramanmaraş Sütçü İmam Üniversitesi
Kahramanmaraş, Türkiye
zirhlioglusamed@gmail.com

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

***Özetçe* —Araçlar hayatımızda çok eskiden beri var olan ve hayatımızı kolaylaştıran icatlardır. Özellikle günümüzde, artık hayatımızın her alanında olmazsa olmazlarımızdandır. Bu göz önüne alınarak bir çok marka ve model üretilmiştir. Bu markalardan birisi de BMW' dir. BMW' nin her yıla özel ve farklı özelliklerde birçok modeli mevcuttur. Bu araçların özellikleri (model, yıl, şanzıman, mil, yakıt tipi, vergi, mesafe başına yakıt tüketimi, motor, fiyat) incelenerek aracın değeri tahmin edilebilir. Bu değerleri incelemek için yapay zeka algoritmalarını (Karar Ağacı, Gradient Boost, XGB, Rastgele Orman Ağacı, LightGBM, CatBoost) kullandık. Kullandığımız algoritmaların tamamı regresyon algoritmalarından oluşmakta. Bunun nedeni ise elimizdeki bağımsız değişkenlerden hareketle, yeni bir bağımlı değişken elde etmek istememizdir.**

Anahtar Kelimeler—Araç, Araç Özellikleri, Karar Ağacı, Gradient Boost, XGB, Rastgele Orman Ağacı, LightGBM, CatBoost

***Abstract*—Vehicles are inventions that have existed in our lives for a long time and make our lives easier. Especially nowadays, it is now an indispensable part of our lives. With this in mind, many brands and models have been produced. One of these brands is BMW. BMW has many models with different features and specific to each year. The value of the vehicle can be estimated by examining the characteristics of these vehicles (model, year, transmission, miles, fuel type, tax, fuel consumption per distance, engine, price). We used artificial intelligence algorithms (Decision Tree, Gradient Boost, XGB, Random Forest Tree, LightGBM, CatBoost) to examine these values. All of the algorithms we use are regression algorithms. The reason for this is that we want to obtain a new dependent variable based on the independent variables we have.**

Keywords—Car, Car Properties, Value, Decision Tree, Gradient Boost, XGB, Random Forest Tree, LightGBM, CatBoost

I. GİRİŞ

Ulaşım, insanlık tarihinin başlangıcından günümüze kadar, insan hayatının en temel ihtiyaçlarından birisi olmuştur. Arabalar icat edilmeden önce ulaşım; at, eşek ve deve gibi hayvanlar kullanılarak sağlanmıştı. Bu durum hem zaman hem de emek kaybına neden olmuştur.

Artan dünya nüfusu ve ihtiyaçlar doğrultusunda ulaşım çok büyük bir önem kazanmıştır. Bu ihtiyaçları gidermek ve ulaşımı daha kolay hale getirmek için yapılan çalışmalar hız kazanmıştır. Bu çalışmalar sonucunda insanlar ihtiyaçları doğrultusunda tarım ve ulaşım araçlarını geliştirmiştir. İlerleyen süreçte ise insanların daha konforlu ve keyifli bir ulaşım deneyimi yaşayabilmesi için çalışmalara devam edilmiştir.

Günümüzde otomobil sektörü, insanlık tarihinin en yaygın sektörlerinden ve yatırım yapılan iş kollarından birisi olmuştur. Başlangıçta sadece ulaşım ve ticaret amaçlı kullanılan otomobiller, günümüzde lüks ve ihtişamlı yaşamın sembolü haline gelmiştir [1].

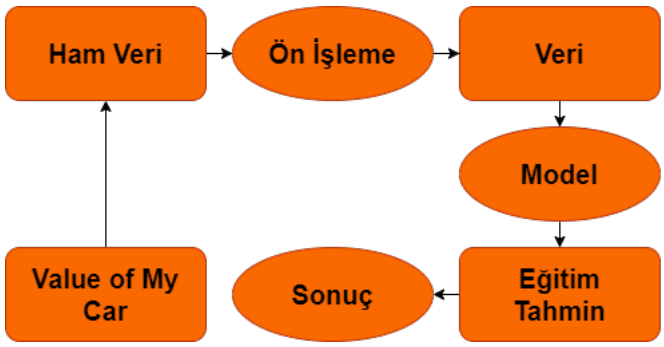
Arabalar uzun ömürlü oldukları için sürekli alınıp satılabilir. İkinci el olarak satılan araçların makul bir şekilde fiyatlandırılması gerekir. İkinci el bir aracın makul fiyat tahmininin yapılması zordur. Bu nedenle, ikinci el araçlar için doğru fiyat tahmin mekanizmasına ihtiyaç vardır [2]. Bu makalede yapay zeka kullanarak ikinci el araçların fiyatlarını tahmin etmeye çalışacağız.

Bu proje için, "100.000 UK Used Car Dataset" isimli veri setini [3] Kaggle'dan temin ettik. Elde ettiğimiz veri seti içerisinden BMW marka aracın verilerini alarak gerekli ön işleme adımlarını yaptık. Veri setimizde 8 adet öznitelik (model, yıl, şanzıman, satış fiyatı, mil, yıllık vergisi, mesafe başına yakıt miktarı ve motor hacmi) bulunmaktadır. Düzenlediğimiz verilerin %30'unu test, %70'ini eğitim verisi olarak ayırdık.

Kullandığımız özniteliklerin özellikleri;

- **Model:** Markanın sınıflara ayırdığı farklı araç tiplerine verdiği isim
- **Yıl:** Aracın üretilip satışa sunulduğu yıl
- **Şanzıman:** Motordan istenilen hareketi şaft veya diferansiyele aktarma organıdır.
- **Satış fiyatı:** Aracın son durumundaki satış fiyatı
- **Mil:** Aracın gittiği toplam yol
- **Yıllık vergi:** Devletin araç kullananlardan düzenli olarak aldığı ücret
- **Mesafe başına yakıt:** Mil başına tüketilen galon miktarı
- **Motor hacmi:** Aracın motor büyüklüğü

Bu veri setini kullanarak ve araçların özniteliklerini göz önünde bulundurarak aracın fiyat tahminini yaptık. Bu fiyat tahmininin daha doğru olması için; Karar Ağacı, Gradient Boost, XGB, Rastgele Orman Ağacı, LightGBM, CatBoost algoritmalarını kullandık. Projeyi Python dilinde, Visual Studio Code editöründe yazdık.



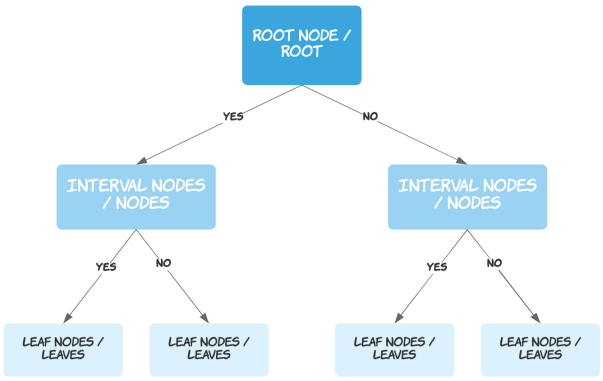
Şekil 1: Value of My Car Akış Diyagramı

Projenin akış diyagramı Şekil 1'de belirtilmiştir. Makalenin devamında kullanılan algoritmalar ile ilgili bilgiler ve sonuç kısmı bulunmaktadır.

II. METODLAR

A. *Decision Tree*

Karar ağaçları, genellikle regresyon ve sınıflandırma problemlerinde kullanılır. Ağaç tabanlı algoritmalardan birisi olup karmaşık veri setlerinde kullanılabilir [4].



Şekil 2: Decision Tree

Kök (root), karar ağaçlarının ilk basamağına(hücresine) verilen isimdir. Problemimizi gözlemlerken, kök kısmındaki koşullar doğrultusunda sınıflandırma yaparız.

Kök hücrelerin altında düğümler (nodes) vardır. Ele aldığımız her bir gözlemi köklerin aracılığı ile sınıflandırırız. Düğüm sayısının artma oranıyla doğru otantılı bir şekilde, elimizde bulunan modelin karmaşıklığı da artar. Bize sonucu veren yapraklardır ve yapraklar ağacın en altında bulunur [4].

Kök hücre seçilirken dikkat edilmesi gereken en önemli etken, veri setini iyi açıklayabilmesidir. Köke karar verirken bizim için önemli bazı değerler vardır [4];

Alt kümenin saflık değerine $Gini$ denir. T, pj, j sınıfının gerçekleşme olasılığını verir. Elimizde bulunan her sınıf için hesaplanır ve elde ettiğimiz sonuçların karesininin toplamını birden çıkartırız. 0 ile 1 arasında bir değere sahiptir ve sonuç 0'a ne kadar yakın olursa o kadar iyi ayrım yapılmıştır denir [4].

$Gini$ indexinin formülü Eşitlik 1'de verilmiştir [15]. Formül değişkenlerinin temsil ettiği veriler de aşağıdaki gibidir;

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

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

Entropy rastgeleliği, belirsizliği ve beklenmeyen durumun ortaya çıkma olasılığını gösterir [5]. Bu olasılığı log_2 tabanında yapar [4].

$$I_H = -\sum_{j=1}^{c} p_j log_2(p_j) \qquad (2)$$

Entropi daha dengeli bir ağaç çıkarmaya meyilli iken $Gini$, frekansı fazla olan sınıfı ayrıştırmaya meyillidir.

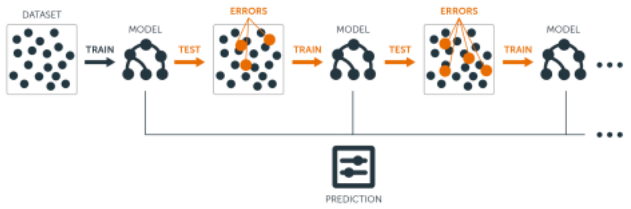
B. *Gradient Boosting Regressor*

Makine öğrenmesi modellerinde doğru tahminleri daha da güçlü bir hale getirmek için Gradient Boosting Regressor kullanılır. Karar ağacı tabanlı algoritmalardan oluşur, yinelemeli olarak çalışırlar. Oluşturulan ağaç yapısı, gelen hatayı en aza indirgeyecek eğilimde olması gerekmektedir. Bu algoritmaları en güçlü kılan bu faktördür [6].

Gradyan yükseltme algoritmasında en temel amaç, elde edeceğimiz maliyet fonksiyonunu en aza indirebilmek için parametreleri tekrarlamaktır. Ağaç eklerken kaybı en aza indirmek için gradyan iniş prosedürü kullanılır [6].

Gradyan yükseltme üç unsur içerir:

- Optimize edilmesi beklenen kayıp fonksiyonun(loss) belirlenmesi.
- Optimize edilmiş veride belirlenen zayıf öğrenci ile tahmin yapmak.
- Loss'u minimum seviyeye indirgemek için zayıf öğrencilere bir katkı modeli eklenir [7].



Şekil 3: Gradyan yükseltme algoritması işleyiş süreci [16]

$$Loss^i = \sum_{j=1}^{n} (Y_j - F^i(X_j))^2 \qquad (3)$$

C. *XGB Regressor*

Gradyan yükseltme algoritmasını çeşitli düzenlemeler ile üstün performanslı haline dönüşmesine denir veya aşırı gradyan artırma olarak da bilinir [9]. Yüksek tahmin gücü elde etmek, boş verileri yönetebilmek ve aşırı öğrenmenin önüne geçmeyi engellemekte oldukça hızlıdır [10].

XGB'nin kullanmanın en temel sebebi bulundurduğu kütüphanenin model performansı ve yürütme hızının yüksek olmasına dayanmaktadır [9].

XGBoost algoritması kullanılırken öncelikle ilk tahmin (base score) yapılmalıdır. Sonraki adımlarda sonuç herhangi bir sayı olabilir. Çünkü yapılacak işlemler yakın sayarak doğru sonuca ulaşacaktır. Sonraki adımda hatalar tahminleyen karar ağacı kurulur. Bu adımdaki amaç hataları öğrenip doğru tahmine yaklaşmaktır. Benzerlik skoru (similarity score) verilerin dallarda ne kadar iyi gruplandığını gösterir. Bu skor oluşturulan ağacın her bir dalı için hesaplanır.

$$SimilarityScore = \frac{Sum of Residuals, Squared}{Number of Residuals + \lambda} \qquad (4)$$

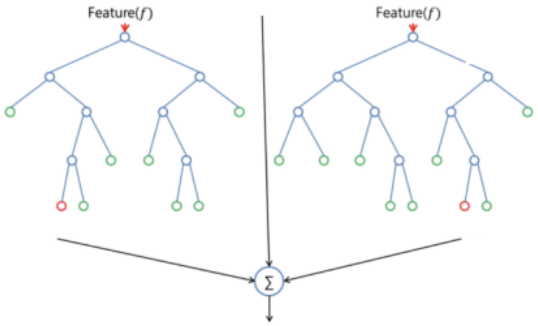
Benzerlik skorlarını hesapladıktan sonra elde ettiğimiz sonuçtan daha iyi bir tahmin yapılıp yapılmayacağını öğrenmek için tüm olasılıklardaki ağaçlar kurulur. Hepsi için ayrı ayrı benzerlik skorları hesaplanır ve hangi ağacın daha etkili yani daha iyi olduğunu düzenlemek için Eşitlik 5'deki kazanç hesabı yapılır [10]. Yukarıda belirttiğimiz benzerlik skorunun sonucu ile dallar değerlendirilirken, elde ettiğimiz Eşitlik 5'deki kazanç değeri ile ağacın tamamı değerlendirilmektedir [10].

$$Kazanc = SolBenzerlikSkoru + SagBenzerlikSkoru - OncekiAgacinBenzerlikSkoru \qquad (5)$$

D. *Random Forest*

Rastgele orman algoritması, denetimli öğrenme algoritması olup hem sınıflandırmada hem de regresyon analizlerinde kullanılır. Aşırı uyumu önler. Random Forest, karar ağaçlarını rastgele bir şekilde oluşturur. İstikrarlı ve doğru bir sonuç tahmini yapabilmek için bunları birleştirir. Ağaç sayısı ve sonuçları arasında doğrudan ilişki bulunmaktadır. Ağaç sayısını ne kadar arttırırsak o kadar kesin bir sonuç elde ederiz [11] [12].

Rastgele orman algoritması, ağaç büyütme aşamasında, normal durumun üzerine rastgelelik durumunu da kullanıyor [12].

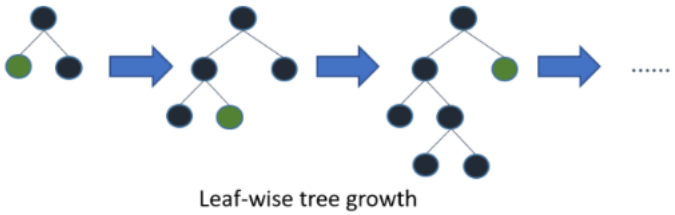


Şekil 4: Random Forest [12]

E. *LightGBM Regressor*

LightGBM, histogram tabanı kullanan bir algoritmadır. Sürekli verileri giriş olarak alır, bu verileri kesikli hale getirir. Bunu yapmasındaki amaç maliyeti azaltmaktır. Karar ağacı algoritmasının bölünmesi ile doğru orantılıdır. Veriyi optimize etmesi, eğitim süresini düşürür ve daha az kaynak kullanılmasını sağlar [13].

Karar ağacı algoritmasında yaprak ve seviye olarak iki farklı strateji kullanılabilir. Yaprak stratejisinde kayıp oranını azaltan yapraklar üzerinden bölünme işlemi devam eder. Seviye stratejisinde hedeflenen durum ise ağaç büyütülürken dengesini korumaktır. Yaprak stratejisinde hatayı minimize etmek amaçlandığı için, seviye stratejisine göre daha düşük hata oranına sahiptir. Bunun etkilediği durumlardan biri de öğrenme aşamasının hızlanmasıdır. Bu yüzden, boosting algoritmalarının geri kalanından ayrılır. Elimizde bulunan veri sayısı ne kadar çoksa, yaprak stratejisini kullanmak o kadar mantıklı olur. Aşırı öğrenmeye sebebiyet vermemek için veri sayısının az olduğu durumlarda kullanıma uygun değildir. Bu durumda da seviye stratejisi kullanılır [13].

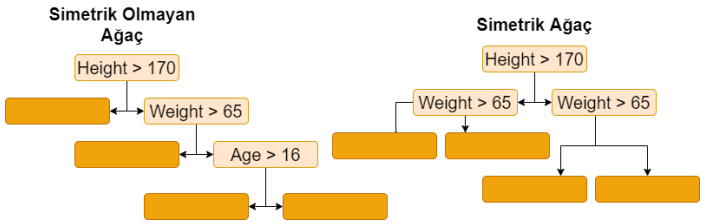


Şekil 5: Yaprak odaklı(leaf wise) strateji [13]

Şekil 6: Seviye odaklı(level wise) strateji [13]

F. *CatBoost Regressor*

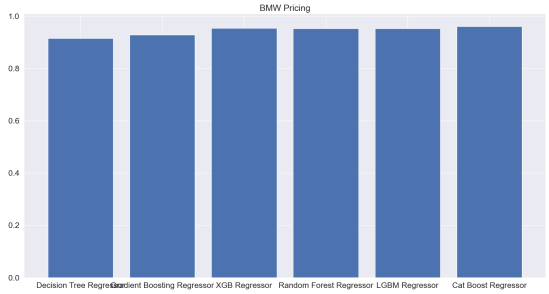
Gradient Boosting tabanlı açık kaynak kodlu bir makine öğrenmesi algoritmasıdır. Veri hazırlığı süresini düşürdüğü için önemli bir yere sahiptir. Elinde bulunan boş veriler ile baş edebilir ancak kategorik veriler için kodlamaya ihtiyaç duyar. Yüksek öğrenme hızı, sayısal, kategorik ve metin verileri ile çalışabilmesi en belirgin özelliklerindendir. Ayrıca aşırı öğrenme sorununu ortadan kaldırmak için simetrik ağaç kurar. Over-fitting (aşırı öğrenme) durumunun oluşması durumunda, belirlenen özelliklere varmadan önce öğrenme işlemi durdurulur [14].



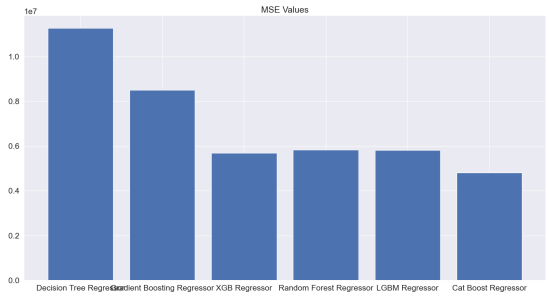
Şekil 7: Simetrik/Simetrik olmayan ağaç

III. SONUÇLAR

Veri setimizdeki verilerin %70'ini eğitim, %30'unu ise test verisi olarak ayırdık. Daha sonra ayırdığımız verileri, algoritmalarımızın eğitim ve tahmin aşamalarında kullandık. Bu aşamalar sonucunda elde ettiğimiz doğruluk değerlerine Şekil 8'de, MSE (ortalama kare hatası) değerlerine de Şekil 9'da yer verdik.



Şekil 8: Tüm Algoritmaların Doğruluk $(r2)$ Değerleri



Şekil 9: Tüm Algoritmaların MSE Değerleri

İkinci el araç fiyatlarının tahmin edilebilmesi için "100.000 UK Used Car" [3] veri setindeki "BMW.csv" verisini aldık. DT, GB, XGB, RF, LGBM ve CB regressor algoritmalarını kullandık. Bu algoritmaların $r2$ ve MSE değerleri Tablo I'de yer almaktadır.

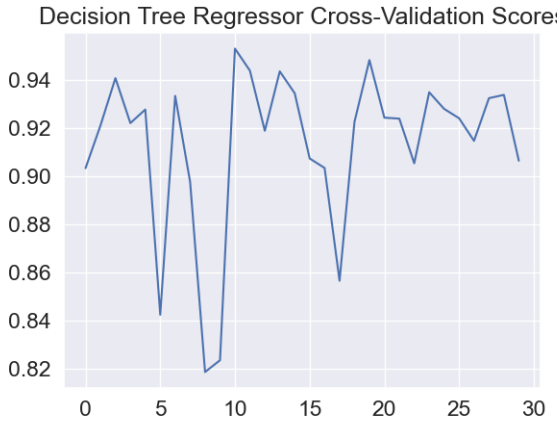
	r2	MSE
DT	0.9209	0.0825
GB	0.9296	0.0657
XGB	0.9538	0.0447
RF	0.9533	0.0453
LGBM	0.9533	0.0449
CB	**0.9614**	**0.0372**

Tablo I: DT, GB, XGB, RF, LGBM ve CB regressor algoritmalarının $r2$ ve MSE degerleri

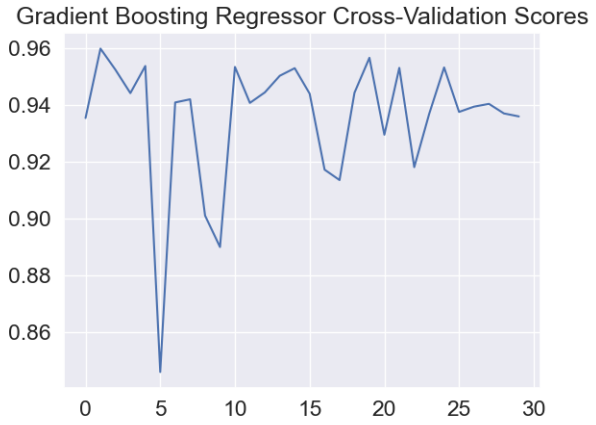
Tablo I'i incelersek, en yüksek doğruluk değerinin $\%96.14$ oranıyla CatBoost algoritmasına ait olduğunu görebiliriz. Aynı zamanda en düşük MSE değeri de 0.0372 oranıyla yine CatBoost'a ait. Bu verilerden en düşük doğruluk değerine sahip olan algoritma ise Decision Tree$(\%92.09)$'dur. $r2$ ve MSE değerlerinin arasındaki ters orantıdan dolayı, en yüksek MSE değeri yine Decision Tree(0.0825)'e aittir. Diğer algoritmaların $r2$ ve MSE değerlerini de Tablo I üzerinde inceleyebiliriz.

A. *Cross-Validation (Çapraz Doğrulama)*

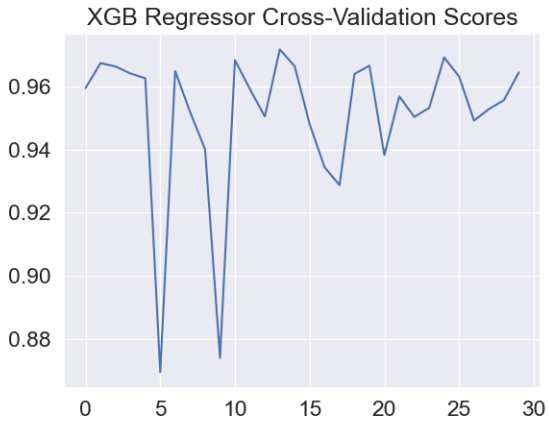
Regresyon algoritmalarımız birer kez eğitildi ve test edildi. Bu eğitimlerin sonuçlarını da Tablo I'de belirttik. Şimdi DT, GB, XGB, RF, LGBM ve CB regressor algoritmalarımızı **Cross-Validation** metoduyla 30 kez rastgele olacak şekilde çalıştırdık ve sonuçları Şekil 10, 11, 12, 13, 14 ve 15'de verdik.



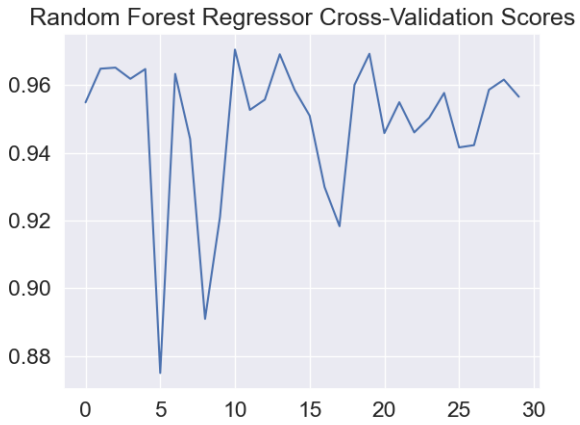
Şekil 10: Decision Tree Regressor Cross-Validation



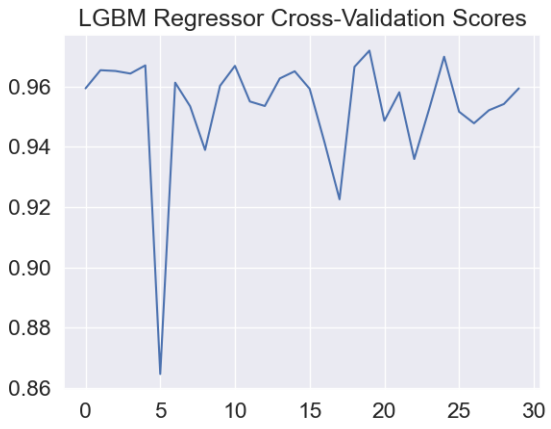
Şekil 11: Gradient Boost Regressor Cross-Validation



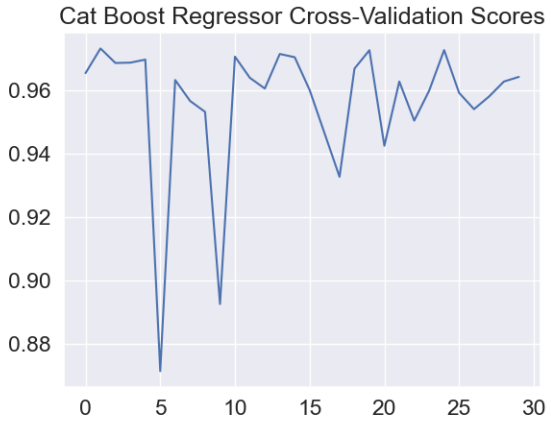
Şekil 12: XGB Regressor Cross-Validation



Şekil 13: Random Forest Regressor Cross-Validation

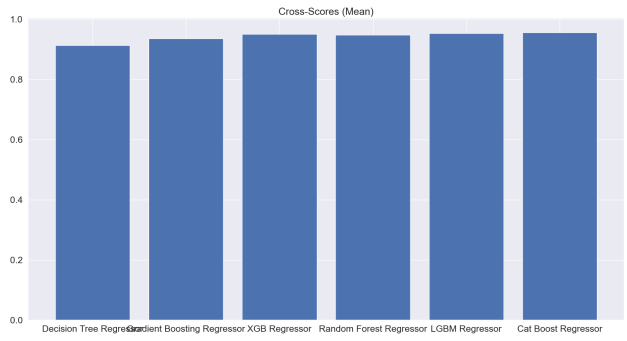


Şekil 14: LGBM Regressor Cross-Validation



Şekil 15: CatBoost Regressor Cross-Validation

Bu algoritmaların Cross-Validaion sonuçlarının ortalamaları Şekil 16'daki gibidir. Bu değerler aynı zamanda Tablo II'de de sayısal olarak yer almakta.



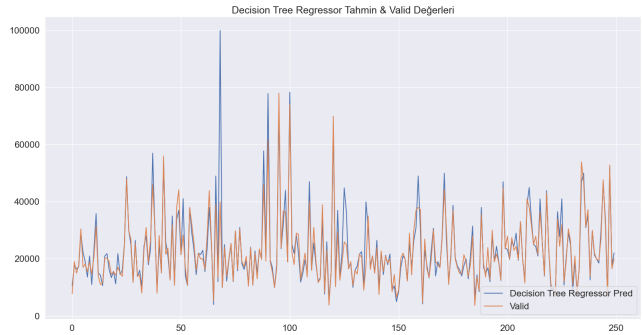
Şekil 16: Cross-Validation Ortalama(Mean) Değerleri

	Cross-Validation Score(mean)
DT	0.9160
GB	0.9356
XGB	0.9512
RF	0.9485
LGBM	0.9531
CB	**0.9562**

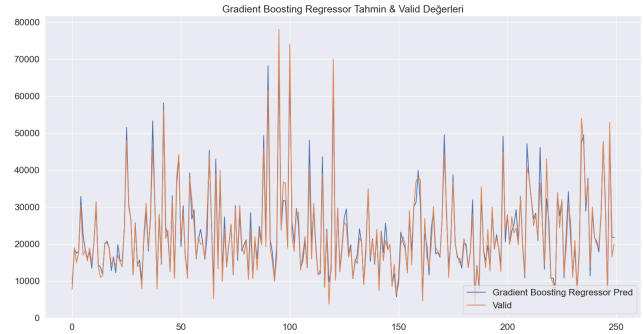
Tablo II: DT, GB, XGB, RF, LGBM ve CB regressor algoritmalarının Cross-Validation sonuçlarının ortalama(mean) degerleri

Tablo II'yi incelersek, en yüksek doğruluk değerinin $\%95.62$ oranıyla CatBoost algoritmasına ait olduğunu görebiliriz. Bu verilerden en düşük doğruluk değerine sahip olan algoritma ise Decision Tree$(\%91.60)$'dır. Diğer algoritmaların Cross-Validation sonuçlarının ortalama değerlerini de Tablo II üzerinde inceleyebiliriz.

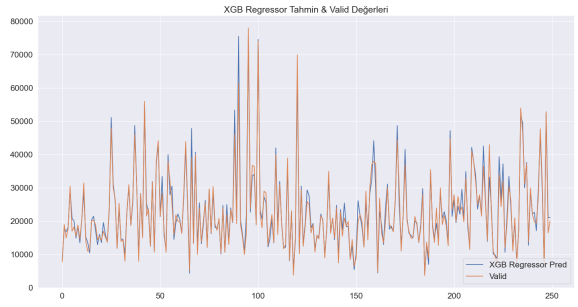
*B. **Valid Data-Pred Data**$(Diff)$*



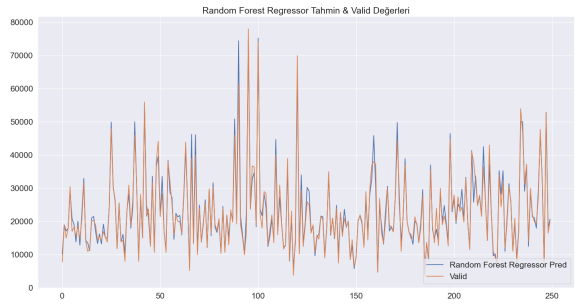
Şekil 17: Decision Tree Regressor Tahmin & Valid Değerleri



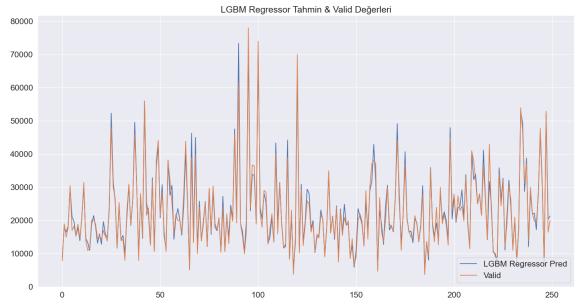
Şekil 18: Gradient Boosting Regressor Tahmin & Valid Değerleri



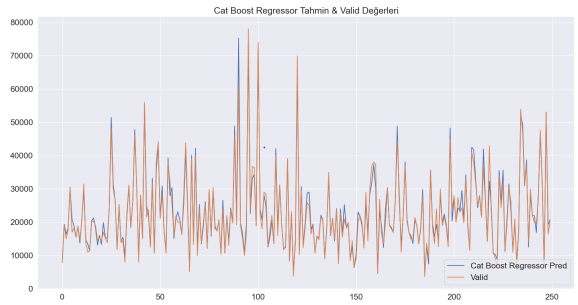
Şekil 19: XGB Regressor Tahmin & Valid Değerleri



Şekil 20: Random Forest Regressor Tahmin & Valid Değerleri



Şekil 21: LGBM Regressor Tahmin & Valid Değerleri

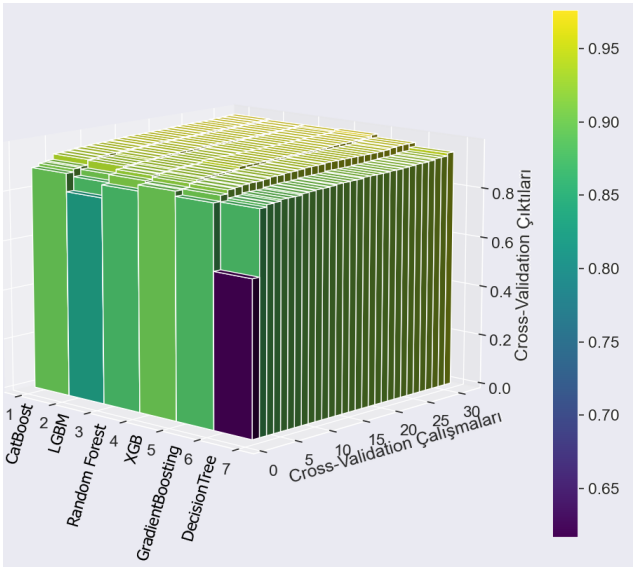


Şekil 22: CatBoost Regressor Tahmin & Valid Değerleri

Şekil 17, 18, 19, 20, 21 ve 22'de de gösterildiği gibi; gerçek $(valid)$ verilerimiz turuncu, algoritmanın tahmin $(predict)$ değerleri ise mavi hatla gösterilmiştir. Bu grafikler oluşturulurken amacımız; tahmin verisinin gerçek veriden ne kadar saptığını(uzaklaştığını) görmektir.

IV. SONUÇ

Bu projede kullandığımız veri seti sayesinde ikinci el araçların, değerinin altında veya üstünde bir fiyata satılması yerine, olması gereken fiyatta satılmasını sağlamak için; yapay zeka kullanarak fiyat tahmini yapmaya çalıştık. Tahmin işlemini, daha önceden belirlenen algoritmaları kullanarak yaptık. Algoritmaları, makalenin önceki başlıklarında da anlattığımız çeşitli işlemlere tâbi tutarak bazı sonuçlar elde ettik. Bu sonuçları karşılaştırdık ve en iyi agoritmayı bulmaya çalıştık. Elimizdeki veri setiyle en uyumlu çalışan algoritmayı bulduk ve raporladık. Eğitim sonuçlarını daha yüksek oranlara taşımak için de veri seti optimize edilebilir, gerekli öznitelikler eklenebilir veya gereksiz olanlar çıkartılabilir. Bu sayede daha doğru bir tahminleme işlemi yapabiliriz.



Şekil 23: Algoritmaların Cross-Validation Sonuçları (3D Grafik)

X eksenine baktığımız zaman algoritmaları, Y eksenine baktığımız zaman algoritmaların çalışma sayılarını görüyoruz. Daha sonradan 3. boyuta ise bu çalışmaların skorları Z ekseni olarak ekleniyor ve Şekil 23'deki 3D grafiği elde ediyoruz. Algoritmalar ve çalışma skorları kendi arasında sıralı olduğu için, sütunlara çapraz açıdan bakıldığında en verimli algoritmanın en yüksekte kaldığını görebiliriz.

KAYNAKÇA

[1] DergiPark - İkinci El Otomobil Talep Fiyatının Regresyon Analizi
dergipark.org.tr/article-file/431769

[2] IjcaOnline - Vehicle Price Prediction
ijcaonline.org/archives/noor-2017-ijca-914373

[3] Kaggle - 100.000 UK Used Car Data Set
kaggle.com/adityadesai13/used-car-dataset-bmw

[4] Medium - Karar Ağaçları
medium.com/deep-learning/karar-ağaçlari

[5] Başkent - Karar Ağacı
baskent.edu.tr/20410964/DM_8.pdf

[6] Data Science - Boosting Algoritmaları
datasciencearth.com/boosting-algoritmalari

[7] Tevfik Bulut - Gradyan Yükseltme Algoritması
tevfikbulut.com/prediction-of-breast-cancer

[8] OpenGenus - Boosting Algorithms
opengenus.org/types-of-boosting-algorithms

[9] OpenGenus - XGBoost
opengenus.org/xgboost

[10] Veri Bilimi Okulu - XGBoost Nasıl Çalışır
veribilimiokulu.com/xgboost-nasil-calisir

[11] Medium - Rastgele Orman Algoritması
medium.com/@cemthecebi/rastgele-orman

[12] DevHunter - Rastgele Orman Algoritması
devhunteryz.wordpress.com/random-forest

[13] Veri Bilimi Okulu - LightGBM
veribilimiokulu.com/lightgbm

[14] Veri Bilimi Okulu - CatBoost Nedir
veribilimiokulu.com/catboost

[15] SlideShare - Random Forest
slideshare.net/SezerFidanc/
random-forest-algoritmasi

[16] BradleyBoehmke - GBM
bradleyboehmke.github.io/gbm.html