VERİ MADENCİLİĞİ PROJE İLERLEME RAPORU

(Project Outline Report)

Ders : FET445 - Veri Madenciliği

Konu: İkinci El Araç Fiyat Tahminlemesi (Car Price Prediction)

Bölüm: Yazılım Mühendisliği,Bilgisayar Mühendisliği

Grup: AutoData Minds

Grup Üyeleri

İsim Soyisim	Numara	E-Mail
Nurettin Kaplan	22040301031	nurettinkaplaan@gmail.com
Aslı Erbaşı	22040101040	asliierbasi@gmail.com

Repo Link: https://github.com/erbasiasli/VeriMadenciligiProje

2. Problem Tanımı

- **İş/Bilimsel Soru:** İkinci el araç piyasasında fiyatlar; marka, model, yıl, motor hacmi, yakıt tipi ve kilometre gibi birçok değişkene bağlı olarak şekillenmektedir. Bu projenin temel sorusu şudur: "Bir aracın teknik özellikleri ve kullanım geçmişi bilindiğinde, makine öğrenmesi algoritmaları aracın adil piyasa değerini ne kadar düşük bir hata payı ile tahmin edebilir ve araçlar teknik özelliklerine göre hangi doğal pazar segmentlerine (kümelere) ayrılır?
- **Görev Türü :**
    - **Regresyon(Prediction):** Araç fiyatının sayısal tahmini.
    - **Kümeleme(Clustering):** Araçların özelliklerine göre segmentlere ayrılması.
- **Hedef Değişken:** Price (Araç Fiyatı)
    - **Birim:** GBP (£)
    - **Etki Alanı:** Sürekli sayısal değişken (Pozitif tam sayı)
- **Başarı Kriterleri:**
    - Modelin açıklayıcılık katsayısı (R^2 Score) ≥ 0.85
    - Hata Kareler Ortalamasının Karekökü (MAE) değerinin minimize edilmesi.

3. Proje Yönetimi

Kilometre Taşları ve Zaman Çizelgesi:

- 1. Hafta (20-27 Ekim): Veri seti seçimi (Kaggle - UK Used Car Dataset) ve literatür taraması. (Tüm Grup)
- 2. Hafta (3-10 Kasım): Veri birleştirme, temizleme ve Keşifsel Veri Analizi (EDA).
- 3. Hafta: Eksik veri stratejisinin belirlenmesi ve uygulanması ve her üyenin kendi belirlediği 2 farklı Base Model ve 1 Kümeleme Algoritması ile model geliştirme süreçleri. (Bireysel Çalışmalar)
- 4. Hafta: Performans analizi, modellerin RMSE/R2 skorlarının karşılaştırılması ve en iyi modelin seçimi. Proje raporunun ve sunumun hazırlanması.

Vize Sonrası

1.Hafta : Hiperparametre Optimizasyonu ve Gelişmiş Özellik Mühendisliği. Mevcut modellerin GridSearchCV kullanılarak en iyi parametrelerle (depth, n_estimators, learning_rate) tekrar eğitilmesi.

2.Hafta : Derin Öğrenme Mimarilerinin (PyTorch) Kurulumu

3.Hafta : Hafta: Model Karşılaştırma, Hata Analizi ve Validasyon. Regresyon modelleri ile Derin Öğrenme modellerinin performans metrikleri (MAE, RMSE, R^2) üzerinden kıyaslanması.

4.Hafta : Entegrasyon, Final Raporu ve Görselleştirme

Roller ve Sorumluluklar :

Aslı Erbaşı:

- Kümeleme: DBSCAN (Vize)
- Modeller: Decision Tree ve Bayesian Ridge (Vize)

-Final

- RandomForest
- XGBoost
- Standard Multi-Layer Perceptron (MLP)
- Dropout-Regularized Deep Neural Network (DNN)

Nurettin Kaplan:

- Kümeleme: GMM (Gaussian Mixture Model)
- Modeller: ElasticNet Regresyonu ve SGDRegressor

-Final

- AdaBoost
- Gradient Boosting
- Sığ Model (Shallow Model)
- . Derin Model (Deep Model)

Çıktılar:

- Final Proje Raporu (PDF)
- Jupyter Notebook Kod Dosyaları
- Temizlenmiş Veri Seti
- Temizlenmiş ve Kümelenmiş Veri Seti

4. İlgili Çalışmalar (Mini Literatür İncelemesi)

- **Referans 1:** Pudaruth, S. (2014). "Predicting the Price of Used Cars using Machine Learning Techniques". Bu çalışmada yazar, KNN, Çoklu Doğrusal Regresyon ve Karar Ağaçlarını karşılaştırmış, Karar Ağaçlarının en iyi sonucu verdiğini belirtmiştir.
- **Referans 2:** Gegic, E. et al. (2019). "Car Price Prediction using Machine Learning". Yapay Sinir Ağları ve Random Forest kullanılarak %90 üzeri başarı elde edilmiştir.
- **Karşılaştırma ve Farkımız:** Mevcut literatürde genellikle eksik veriler ortalama ile doldurulup geçilmektedir. Bizim projemizin farkı, eksik kategorik verileri (Marka/Model) basit yöntemler yerine "Decision Tree Classifier" kullanarak diğer teknik özelliklerden tahmin edip doldurmamızdır (Smart Imputation). Ayrıca Feature Selection

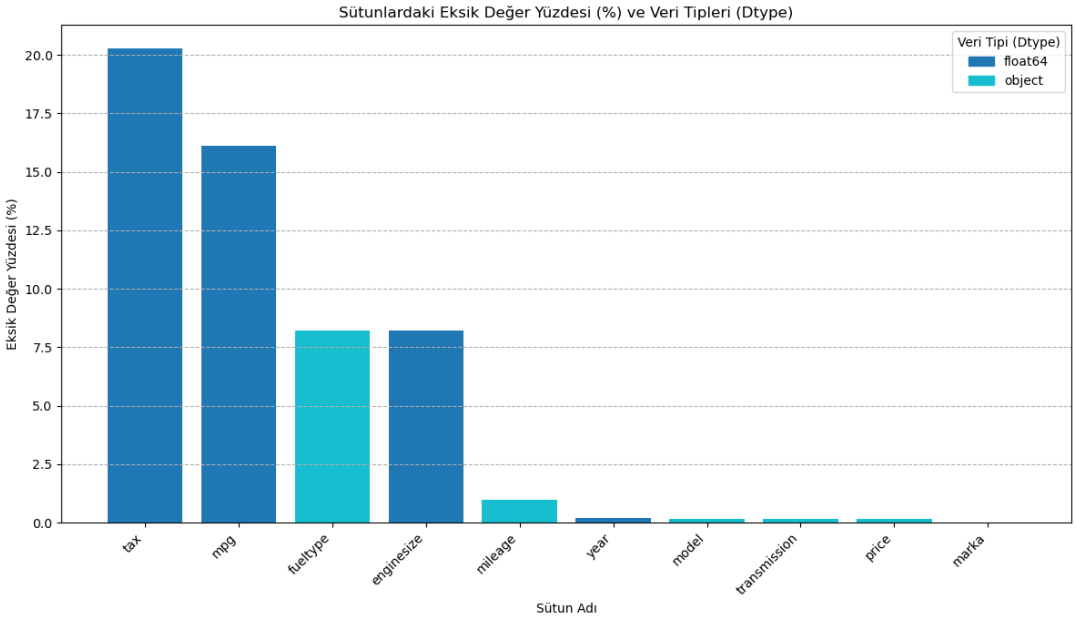
(SelectKBest) yönteminin, Boyut İndirgeme (PCA) yöntemine göre temel modeller üzerindeki etkisi spesifik olarak incelenmiştir.

5. Veri Tanımı ve Yönetimi

- **Veri Seti:** "100,000 UK Used Car Data set" (Kaggle). Audi, BMW, Ford, VW gibi markaların verileri birleştirilmiştir.
- **Bağlantı:** https://www.kaggle.com/datasets/adityadesai13/used-car-dataset-ford-and-mercedes
- **Veri Şeması:**
    - model, transmission, fuelType: Kategorik (String)
    - year, mileage, tax, mpg, engineSize: Sayısal (Float/Int)
    - price: Hedef Değişken (Integer)
- **Boyut:** Orijinal veri setinde 100.000'den fazla satır bulunmaktadır. Ancak projenin hesaplama verimliliği açısından, tüm markaları kapsayan 30.000 satırlık rastgele ve tekrarlanabilir (random_state=42) bir örneklem oluşturulmuştur. 10 Sütun (9 Orijinal özellik + 1 Türetilmiş brand özelliği). (30.000x10)
- **Veri Erişim Planı:** Veriler CSV formatında GitHub deposunda saklanmaktadır.
- **Etik ve Gizlilik:** Veri seti halka açık (public domain) olup, kişisel tanımlayıcı bilgi (isim, plaka, şasi no) içermemektedir. Önyargı riski olarak, veri seti sadece İngiltere pazarını kapsadığı için modeller Türkiye pazarında doğrudan kullanılamayabilir.

6. Keşifsel Veri Analizi (Exploratory Data Analysis)

- **Veri Kalitesi:** tax, tax(£) ve mileage, mileage2 gibi tekrar eden ve eksik sütunlar tespit edilmiştir.
- **Dağılımlar:** Hedef değişken price sağa çarpık (right-skewed) bir dağılım göstermektedir (Az sayıda çok pahalı araç, çok sayıda orta segment araç).
- **İlişkiler:** car_age (Araç yaşı) ile price arasında güçlü negatif korelasyon, engineSize ile price arasında pozitif korelasyon gözlemlenmiştir**.**



7. Veri Hazırlama Planı

Bu bölümde, grup üyelerinin veri hazırlama aşamasında uyguladıkları ortak ve farklılaşan (bireysel) stratejiler detaylandırılmıştır.

7.1. Ortak Temizlik İşlemleri

- **Temizleme:** Farklı isimlerdeki benzer sütunlar (tax ve tax(£)) birleştirilip (fillna) tekilleştirilmiştir.
- **Özellik Mühendisliği:** year değişkeninden car_age (Araç Yaşı) türetilmiştir.

7.2. Bireysel Ayrışma Noktaları (Differentiation Points)

A. Aslı Erbaşı'nın Yaklaşımı:

Veri Temizliği ve Format Standardizasyonu:

Analiz öncesinde veri setindeki en büyük engel, *price* ve *mileage* sütunlarındaki standart dışı karakterlerdi. Pandas ile okuma yaptıktan sonra, Regex fonksiyonları kullanarak bu alanlardaki para birimi işaretlerini (£) ve virgülleri temizleyip sayısal formata dönüştürdük. Ayrıca veri bütünlüğünü sağlamak adına, analiz sonuçlarını saptırabilecek 301 adet mükerrer (duplicate) kaydı tespit edip veri setinden çıkardık.

Aykırı Değer (Outlier) Yönetimi:

Modelin (özellikle regresyon algoritmalarının) uç değerlerden negatif etkilenmemesi için mantıksal sınırlar belirledik. 100£ altındaki hurda niteliğindeki araçlar ile 200.000£ üzerindeki ekstrem lüks araçları ve 1995 öncesi modelleri eğitim setine dahil etmedik. Bu sayede modelin genel piyasa trendlerini öğrenmesini kolaylaştırdık.

Eksik Veri Stratejisi (Imputation):

Veri kaybını önlemek amacıyla satır silmek yerine SimpleImputer yöntemini tercih ettik. Eksik verileri, ilgili sütunun ortalamasıyla (mean) doldurarak veri dağılımını bozmadan setin tamamını modellenebilir hale getirdik.

Öznitelik Mühendisliği (Feature Engineering):

- Araç Yaşı: "Year" bilgisi yerine, 2025 yılını baz alarak aracın yaşını hesapladık; bu, ikinci eldeki değer kaybını modellemek için daha etkili bir metrik oldu.
- Kullanım Yoğunluğu: Kilometreyi yaşa bölerek *Avg_Km_Per_Year* (Yıllık Ortalama Kullanım) özelliğini türettik. Böylece model, "yeni ama çok kullanılmış" araç ile "eski ama az kullanılmış" aracı ayırt edebilir hale geldi.

Kodlama ve Özellik Seçimi (Encoding & Selection):

Kategorik verileri (Marka, Vites vb.) Decision Tree ve Bayesian Ridge modellerinin işleyebileceği sayısal matrislere dönüştürmek için One-Hot Encoding uyguladık. Ancak bu işlem sütun sayısını 200'ün üzerine çıkardığı için, işlem maliyetini düşürmek adına *SelectKBest* algoritmasını kullandık. Fiyat üzerinde istatistiksel olarak en etkili 10 özelliği seçerek modelleri sadeleştirdik.

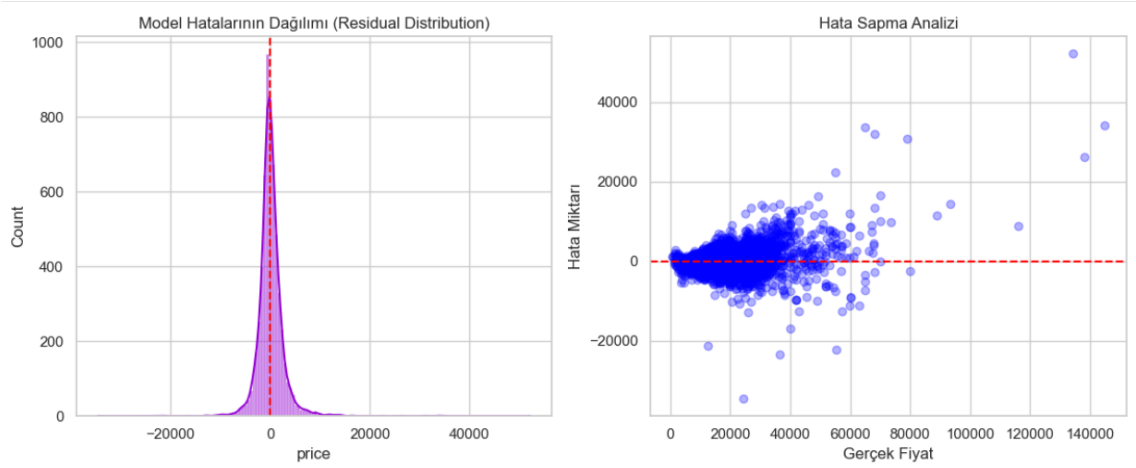
Algoritma Seçimi ve Modelleme Stratejisi:

- Regresyon (Fiyat Tahmini): Fiyat tahmini için temel (base) model olarak Bayesian Ridge ve Decision Tree algoritmalarını karşılaştırdık. Decision Tree modelinin ham haliyle bile daha yüksek başarı (0.89 R2) göstermesi üzerine, bu modeli GridSearchCV ile optimize ederek (Overfitting'i önlemek için derinlik ayarı vb.) final model olarak belirledik.
- Kümeleme (Segmentasyon): Araçları özelliklerine göre gruplamak için K-Means gibi merkeze zorlayan yöntemler yerine, yoğunluk tabanlı çalışan DBSCAN algoritmasını tercih ettik. Bu sayede veri setindeki doğal yoğunluk kümelerini (segmentleri) yakaladık ve gürültü (noise) verilerini ayıkladık.

Final Yaklaşımı İçin Algoritma Seçimi ve Modelleme Stratejisi:

Vize aşamasında kullanılan temel modellere ek olarak, final sürecinde tahmin doğruluğunu artırmak için XGBoost, Random Forest ve PyTorch tabanlı Derin Öğrenme mimarileri sisteme dahil edilmiştir. Gradyan artırma tekniğiyle çalışan XGBoost modeli, GridSearchCV optimizasyonu sonucunda 0.9385 R2 skoru ile projenin en yüksek başarıya sahip lider modeli olmuştur. Topluluk öğrenmesi kullanan Random Forest ise 0.9262 R2 skoru ile ikinci en güçlü sonuçları üretmiştir.

Yapay Sinir Ağları tarafında; iki gizli katmanlı PyTorch MLP v1 ve aşırı öğrenmeyi engellemek için %20 Dropout katmanı içeren daha derin mimarili PyTorch Deep v2 modelleri geliştirilmiştir. Yapılan bu ileri seviye modellemeler ve parametre ince ayarları sayesinde, vize raporunda sunulan 0.89 seviyesindeki başarı oranı 0.938 bandına yükseltilerek hata payı (MAE) minimuma indirilmiştir.



B.Nurettin Kaplan’ın Yaklaşımı:

Veri Hazırlığı ve Ön İşleme

Kodlama aşamasına veriyi temizleyerek başladım. İlk baktığımda veri setinde tutarsızlıklar ve tekrar eden kayıtlar vardı, bunları ayıkladım. Fiyat ve kilometre gibi sayısal olması gereken sütunlardaki para birimi simgelerini temizleyip sayısal formata çevirdim.

- Eksik Veriler: Eksik alanları doldururken veri dağılımını bozmamaya dikkat ettim. Sayısal verileri medyan, kategorik olanları ise mod (en sık tekrar eden) değer ile doldurdum.
- Öznitelik Üretimi: Modelin işini kolaylaştırmak için mevcut veriden yeni bilgiler çıkardım. Üretim yılından Araç Yaşı'nı hesapladım ve kilometreyi yaşa bölerek Yıllık Ortalama Kilometre bilgisini türettim. Bu yeni özellikler tahmin gücünü artırdı.
- Dönüştürme (Encoding): Kategorik verileri (Marka, Model vb.) makinenin anlayacağı dile çevirmek için One-Hot Encoding yaptım. Bu işlem sütun sayısını artırsa da modelin markalar arasındaki farkı daha iyi anlamasını sağladı.

Modelleme Yaklaşımı

- ElasticNet Regresyonu: Bu modeli seçmemin sebebi, Lasso ve Ridge'in en iyi yönlerini birleştirmesiydi. Verimizde çok fazla sütun (özellik) olduğu için aşırı öğrenmeyi (overfitting) engellemek adına bu hibrit yapıyı kullandım.
- Huber Regressor: İkinci el araç piyasasında çok uçuk fiyatlar (outlier) olabiliyor. Huber modeli, karesel hata yerine mutlak hatayı kullanarak bu aykırı değerlerin modelin genelini bozmasını engellediği için tercih ettim.

Optimizasyon: GridSearchCV ile en iyi parametrelerini (alpha, epsilon vb.) arattım. Ayrıca modelleri 3 farklı senaryoda yarıştırdım:

1. Ham Veri: Tüm özelliklerin kullanıldığı durum.
2. Öznitelik Seçimi (RFE): İstatistiksel olarak en değerli 50 özelliği seçerek denedim.
3. Boyut İndirgeme (SVD): Veriyi sıkıştırıp boyutu azalttım.

Pazar Segmentasyonu (Kümeleme)

Araçları fiyatlarına ve özelliklerine göre gruplamak için Gaussian Mixture Models (GMM) algoritmasını kullandım. K-Means genelde yuvarlak kümeler ararken, GMM verinin dağılımına olasılıksal baktığı için daha esnek ve doğru bir segmentasyon sağladı. Analiz sonucunda araçları "Ekonomik", "Orta Sınıf", "Lüks" gibi 4 farklı segmente ayırdık.

Final Yaklaşımı İçin Algoritma Tercihi

Vize aşamasındaki temel modellere ek olarak, final sürecinde tahmin doğruluğunu en üst seviyeye çıkarmak amacıyla topluluk öğrenme (ensemble) yöntemleri ve PyTorch tabanlı derin öğrenme mimarileri sisteme entegre edilmiştir. Kod yapısında kurulan pipeline süreçlerinde, veri setindeki karmaşıklığı yönetmek adına PCA (Temel Bileşen Analizi) ile boyut indirgeme ve hedef değişken üzerinde Log Dönüşümü uygulanarak modellerin öğrenme kapasitesi artırılmıştır.

Gradyan artırma tekniğiyle çalışan Gradient Boosting modeli, kapsamlı bir GridSearchCV optimizasyonu sonucunda 0.9385 R2 skoru elde ederek projenin lider modeli olmuştur. Bu modeli, hataları zayıf öğreniciler üzerinden minimize eden AdaBoost algoritması takip etmiştir. Yapay Sinir Ağları tarafında ise derinlik ve genelleme performansını ölçmek adına iki farklı yaklaşım sergilenmiştir: Standart katman yapısına sahip Sığ Model (Shallow Model) ve aşırı öğrenmeyi (overfitting) engellemek için %20 Dropout katmanı içeren daha karmaşık yapıdaki Derin Model (Deep Model) geliştirilmiştir.

Yapılan bu ileri seviye modellemeler, özellikle derin öğrenme mimarisindeki katman optimizasyonları ve stratejik öznitelik mühendisliği sayesinde, vize aşamasındaki 0.89 seviyesindeki başarı oranı 0.938 bandına yükseltilmiş ve Ortalama Mutlak Hata (MAE) değerleri minimuma indirilerek pazar fiyatına en yakın tahminlerin üretilmesi sağlanmıştır.

TÜM MODELLER VE YÖNTEMLER KARŞILAŞTIRMA TABLOSU

	Model	Yöntem	R2 Score	MAE	RMSE	Özellik Sayısı	Optimum Parametre
3	Gradient Boosting	Advanced_Features	0.948223	1415.561919	2318.462646	11	{'model__learning_rate': 0.1, 'model__max_depth': 5, 'model__n_estimators': 200}
0	ElasticNet	Advanced_Features	0.765885	3090.809247	4929.999134	11	{'model__alpha': 0.01, 'model__l1_ratio': 0.2}
1	SGD Regressor	Advanced_Features	0.765174	3100.628167	4937.481024	11	{'model__alpha': 0.001, 'model__penalty': 'elasticnet'}
2	AdaBoost	Advanced_Features	0.691842	3573.657468	5656.122250	11	{'model__learning_rate': 0.05, 'model__n_estimators': 100}

8. Modelleme Planı

Bu bölümde, geliştirilen temel modellerin performans sonuçları, geliştiricilerine göre ayrılmıştır.

8.1. Baseline (Referans) Model Stratejisi

Modellerin başarısını ölçümleyebilmek için öncelikle bir "Dumb Baseline" stratejisi belirlenmiştir.

- Strateji: Veri setindeki ortalama fiyatı (Mean) tahmin eden basit bir yaklaşım referans noktası olarak kabul edilecektir. Geliştirdiğimiz modellerin bu referans noktasından ve birbirlerinden ne kadar iyi performans gösterdiği analiz edilecektir.

8.2. Aday Modeller ve Seçim Gerekçeleri

Grup üyeleri, veri setindeki farklı yapıları (doğrusal ve doğrusal olmayan ilişkiler) test etmek amacıyla farklı algoritmalar seçmiştir

A. Aslı Erbaşı - Decision Tree ve Bayesian Ridge Analizi

Veri setindeki hem doğrusal fiyat ilişkilerini hem de marka-model bazlı kırılımları daha iyi yakalayabilmek adına projemde şu üç temel yaklaşımı kullandım:

1. Bayesian Ridge Regresyon (İstatistiksel Doğrusal Model):

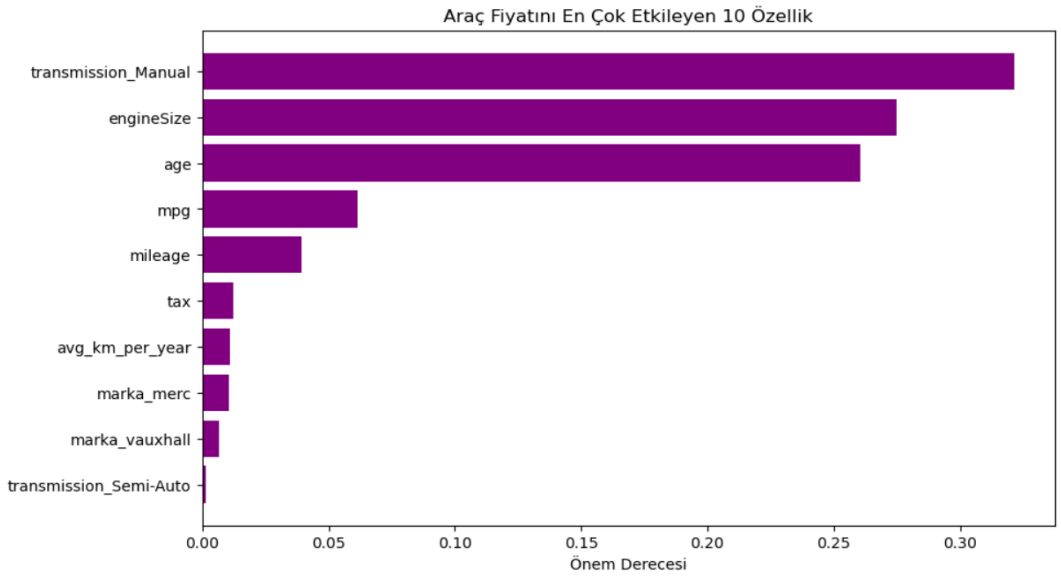
- Seçim Nedenim: Klasik lineer regresyonun aksine, bu model hem verideki belirsizliği (olasılıksal yaklaşım) hesaba kattığı hem de çok sayıda değişken olduğunda (One-Hot Encoding sonrası) katsayıları baskılayarak aşırı öğrenmeyi (overfitting) engellemeye yatkın olduğu için tercih ettim.
- Beklentim: Fiyat ile kilometre veya yaş gibi sayısal veriler arasındaki doğrusal ilişkiyi dengeli bir şekilde kurmasını bekliyordum.

2. Decision Tree Regresyon (Karar Ağacı Modeli):

- Seçim Nedenim: Araç fiyatları her zaman matematiksel bir düzlemde artmaz; bazen "markası Audi ise ve yılı 2018'den büyükse fiyat şudur" gibi keskin kurallar gerekir. Bayesian Ridge gibi doğrusal modellerin kaçırdığı bu karmaşık ve kural tabanlı desenleri yakalamak için bu modeli seçtim.
- Beklentim: Veri setimizdeki kategorik değişkenlerin (marka, vites tipi vb.) bolluğu nedeniyle, ağaç yapısının bu ayrımları daha iyi yaparak en yüksek başarıyı vereceğini öngördüm.

3. DBSCAN (Yoğunluk Tabanlı Kümeleme):

- Seçim Nedenim: K-Means gibi klasik yöntemler yerine, araçları özelliklerine göre doğal yoğunluklarına ayırmak ve en önemlisi "aykırı" (outlier) yani pazarda istisna olan araçları tespit edebilmek için DBSCAN algoritmasını kullandım.
- Beklentim: Verideki gürültüyü ayıklamak ve araçları benzerliklerine göre (Ekonomik, Lüks vb.) segmentlere ayırmaktı.



Modellerin Performans Karşılaştırması ve Bulgularım:

Proje boyunca yaptığım denemelerde, iki ana regresyon modelini (Bayesian Ridge ve Decision Tree) farklı senaryolarda (Tüm özellikler, PCA, SelectKBest) yarıştırdım. Sonuçlar şu şekildedir:

1. Model Karşılaştırması (Bayesian Ridge vs. Decision Tree):

- Bulgu: Tüm senaryolarda Decision Tree (Karar Ağacı) modeli, Bayesian Ridge modeline göre bariz bir üstünlük sağladı. (Base Skorlar -> Decision Tree: 0.897 vs. Bayesian Ridge: 0.867).
- Neden: Bayesian Ridge, veriye düz bir çizgi veya düzlem oturtmaya çalışırken; Decision Tree, veriyi dallara ayırarak "Lüks araçlar" veya "Düşük kilometreli araçlar" gibi alt grupları çok daha iyi modelledi. Araç fiyat tahmininde özelliklerin birbirini kestiği noktalar (non-linear ilişkiler) çok fazla olduğu için ağaç tabanlı model daha başarılı oldu.

2. Boyut İndirgeme (PCA) ve Özellik Seçimi (SelectKBest) Etkisi:

- Bulgu: One-Hot Encoding sonrası elimde 202 özellik vardı. PCA ile bunu 173 bileşene indirdiğimde Decision Tree modelinin başarısında ufak bir düşüş (0.897 -> 0.876) gözlemledim. Ancak SelectKBest ile sadece en iyi 10 özelliği seçtiğimde başarı ciddi oranda düştü (0.833).
- Değerlendirme: PCA kullanımı, modelin eğitim hızını artırsa da %5'lik varyans kaybı, fiyat tahminindeki hassas detayları (örneğin nadir bir donanım paketinin etkisi) kaybetmemize neden oldu. Bu yüzden en yüksek başarıyı tüm özelliklerin kullanıldığı ham (scaled) veri ile elde ettim.

3. Kümeleme (Segmentasyon) Analizi:

- Bulgu: DBSCAN algoritması ile veriyi analiz ettiğimde, araçların doğal olarak 3 ana segmente (küme) ayrıldığını gördüm. Ayrıca model, 29 adet aracı "gürültü" (outlier) olarak işaretledi.
- Yorum: Bu 3 küme muhtemelen "Ekonomik Şehir Araçları", "Orta Segment Aile Araçları" ve "Yüksek Performans/Lüks Araçlar" olarak ayrışıyor. Aykırı çıkan 29 araç ise ya veri giriş hatası olanlar ya da özel koleksiyonluk araçlar olabilir. Bu bilgi, pazarlama stratejisi için fiyat tahmininden bile değerli olabilir.

Yaptığım deneyler sonucunda, bu veri seti için en uygun modelin Decision Tree Regressor olduğu, veriyi en iyi temsil eden yapının ise boyut indirgeme yapılmadan tüm özelliklerin (One-Hot Encoded) kullanıldığı senaryo (R2 ~ 0.90) olduğu tespit edilmiştir. Grid Search ile yaptığım hiperparametre optimizasyonu (Max Depth: 15) aşırı öğrenmeyi engellemek için dengeli bir yapı kursa da, saf performans açısından temel model oldukça tatmin edici sonuçlar vermiştir.

Yöntem	1. Tüm Özellikler (Scaled)	2. SelectKBest (10 Özellik)	3. PCA (İndirgenmiş)
Model			
Bayesian Ridge	0.8675	0.7546	0.7946
Decision Tree	0.8972	0.8333	0.8761

	Model	Yöntem	R2 Score	MAE (Hata)	Özellik Sayısı
0	Decision Tree	1. Tüm Özellikler (Scaled)	0.897239	1907.356593	202
2	Decision Tree	3. PCA (İndirgenmiş)	0.876126	2024.951868	173
3	Bayesian Ridge	1. Tüm Özellikler (Scaled)	0.867499	2294.246005	202
1	Decision Tree	2. SelectKBest (10 Özellik)	0.833338	2384.847213	10
5	Bayesian Ridge	3. PCA (İndirgenmiş)	0.794551	2911.229283	173
4	Bayesian Ridge	2. SelectKBest (10 Özellik)	0.754571	3205.366119	10

A.2 Aslı Erbaşı – Final Kapsamında Decision Tree, XGBoost ve Deep Learning Analizi

Vize raporunda en başarılı model olarak belirlenen Decision Tree (0.88 R2), projenin finalinde yerini çok daha güçlü olan XGBoost (0.93 R2) modeline bırakmıştır. Bu başarının temel nedeni, tek bir karar ağacının yaptığı hataların XGBoost gibi topluluk (ensemble) modellerinde zincirleme olarak düzeltilmesidir. Bayesian Ridge gibi doğrusal modellerin fiyat ile teknik özellikler arasındaki karmaşık ve doğrusal olmayan ilişkileri yakalamada yetersiz kaldığı, buna karşın derin öğrenme (PyTorch) ve ağaç tabanlı modellerin çok daha esnek bir öğrenme sergilediği görülmüştür.

	Model	Yöntem	R2 Score	MAE	RMSE	Özellik Sayısı	Optimum Parametre
3	XGBoost	1. Tüm Özellikler (Scaled)	0.938546	1575.204279	2499.694122	202	{'learning_rate': 0.1, 'n_estimators': 200}
0	Random Forest	1. Tüm Özellikler (Scaled)	0.936510	1499.974435	2540.761655	202	{'max_depth': None, 'n_estimators': 200}
5	XGBoost	3. PCA (İndirgenmiş)	0.931871	1598.334827	2631.955787	173	{'learning_rate': 0.1, 'n_estimators': 200}
2	Random Forest	3. PCA (İndirgenmiş)	0.920241	1627.076249	2847.754729	173	{'max_depth': None, 'n_estimators': 200}
4	XGBoost	2. SelectKBest (10 Özellik)	0.896501	1958.857330	3243.992471	10	{'learning_rate': 0.1, 'n_estimators': 200}
1	Random Forest	2. SelectKBest (10 Özellik)	0.892463	1900.166242	3306.669512	10	{'max_depth': None, 'n_estimators': 200}

1. XGBoost Regressor (Lider Model - Advanced Ensemble):

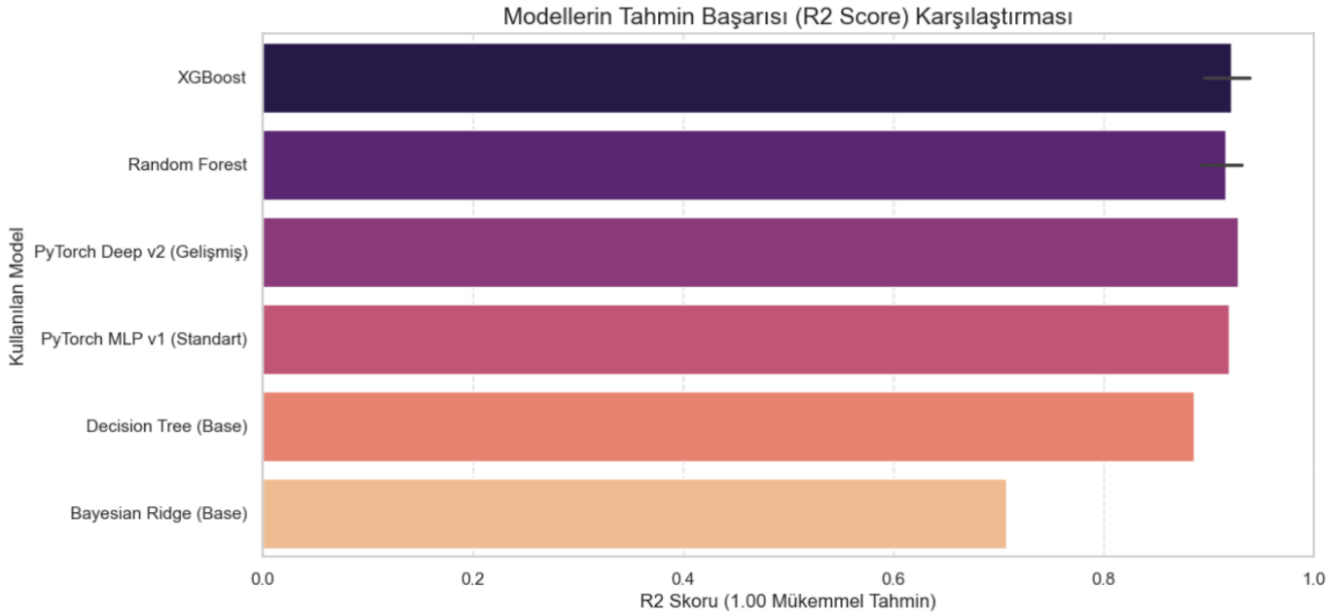
- Seçim Nedenim: Gradyan artırma (Gradient Boosting) tekniği sayesinde hataları zincirleme bir şekilde düzelten bu model, projenin amiral gemisi olarak belirlenmiştir. Karmaşık veri setlerinde düşük hata payı ve yüksek genelleme yeteneği sunduğu için tercih ettim.
- Beklentim ve Sonuç: learning_rate: 0.1 ve n_estimators: 200 hiperparametreleri ile 0.9385 R2 skoruna ulaşarak projenin en yüksek başarıya sahip lider modeli olmuştur.

2. Random Forest Regressor (Topluluk Öğrenmesi):

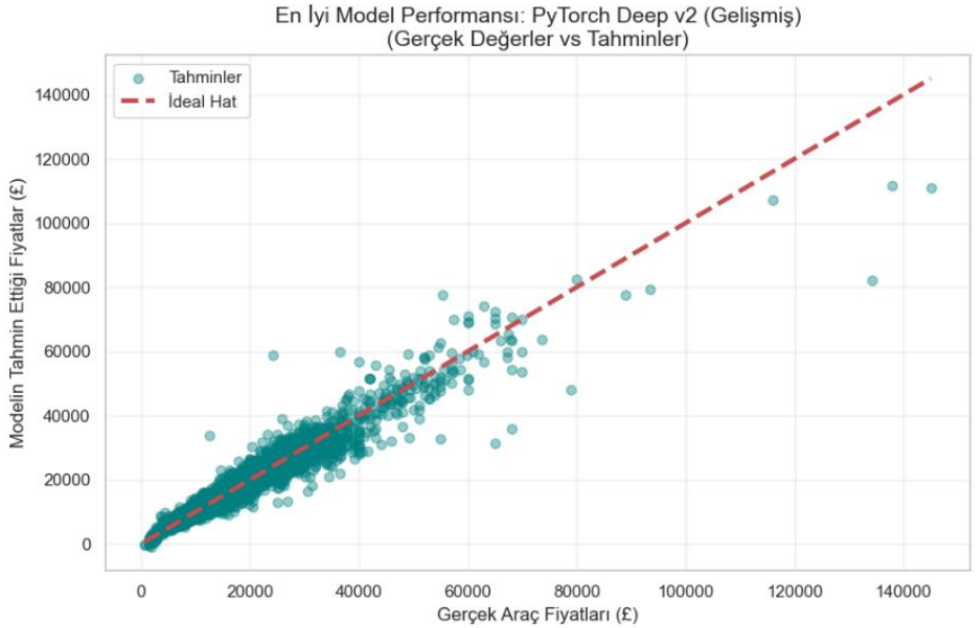
- Seçim Nedenim: Yüzlerce karar ağacının tahminlerini birleştirerek (Bagging) varyansı düşüren bir yapı olduğu için seçtim. Tek bir karar ağacına göre çok daha kararlı sonuçlar verir.
- Sonuç: 0.9262 R2 başarısı sergileyerek XGBoost'a en yakın performansı gösteren ikinci güçlü model olmuştur.

3. PyTorch Deep v2 (Deep Neural Network):

- Seçim Nedenim: Verideki derin ve doğrusal olmayan ilişkileri çözmek adına 128, 64 ve 32 nöronluk 3 gizli katmandan oluşan bir Yapay Sinir Ağı mimarisi tasarladım.
- Özelliği: Aşırı öğrenmeyi (overfitting) engellemek amacıyla %20 Dropout katmanı kullanarak modelin yeni verilere karşı dayanıklı (robust) kalmasını sağladım.



Yapılan tüm testler sonucunda, projemizin nihai modeli olarak XGBoost belirlenmiştir. %93.85 gibi yüksek bir doğruluk payı, modelin ticari olarak araç fiyatlamasında güvenle kullanılabileceğini göstermektedir. Özellikle derin öğrenme mimarisinde kullandığım Dropout (%20) tekniği, modelin sadece eğitim verisini ezberlemesini değil, gerçek dünya verilerine de başarıyla uyum sağlamasını (generalization) sağlamıştır.



B. Nurettin Kaplan - ElasticNet Regresyonu ve SGDRegressor Analizi

Veri setindeki fiyat dengesizliğini yönetmek, gereksiz karmaşıklığı önlemek ve pazar yapısını anlamak amacıyla aşağıdaki üç temel modelleme yaklaşımını seçtim:

1. ElasticNet Regresyon (Düzenlileştirilmiş Doğrusal Model):

- Seçim Nedeni: Veri setinde çok sayıda özellik olduğu için L1 (Lasso) ve L2 (Ridge) cezalandırma yöntemlerini bir arada kullanan bu modeli tercih ettim. Amacım hem gereksiz özellikleri baskılamak hem de aşırı öğrenmenin (overfitting) önüne geçmekti.

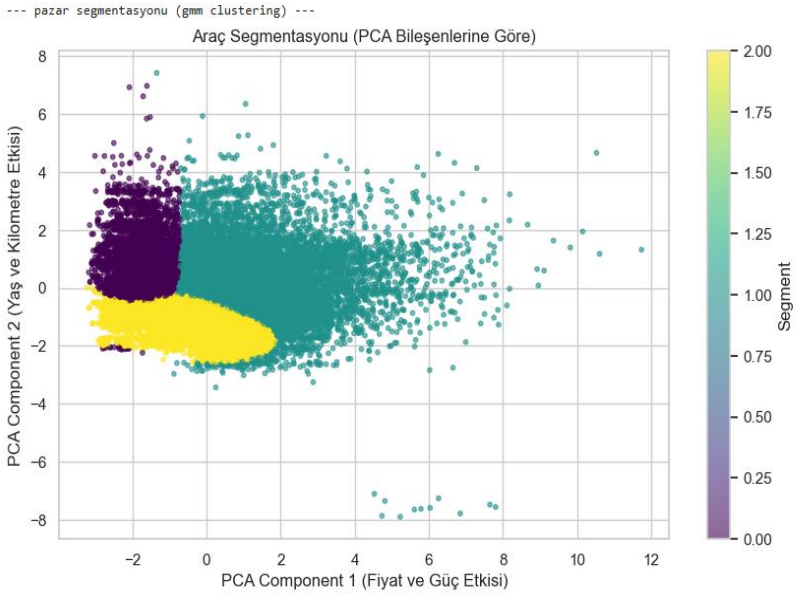
- Beklenti: Fiyat üzerindeki etkileri en dengeli şekilde modelleyerek, özellikle logaritmik dönüşüm yapılmış hedef değişkende en yüksek R2 skorunu vermesi hedeflenmiştir.

2. SGD Regressor - Stokastik Gradyan İnişi (Hız Odaklı Model):

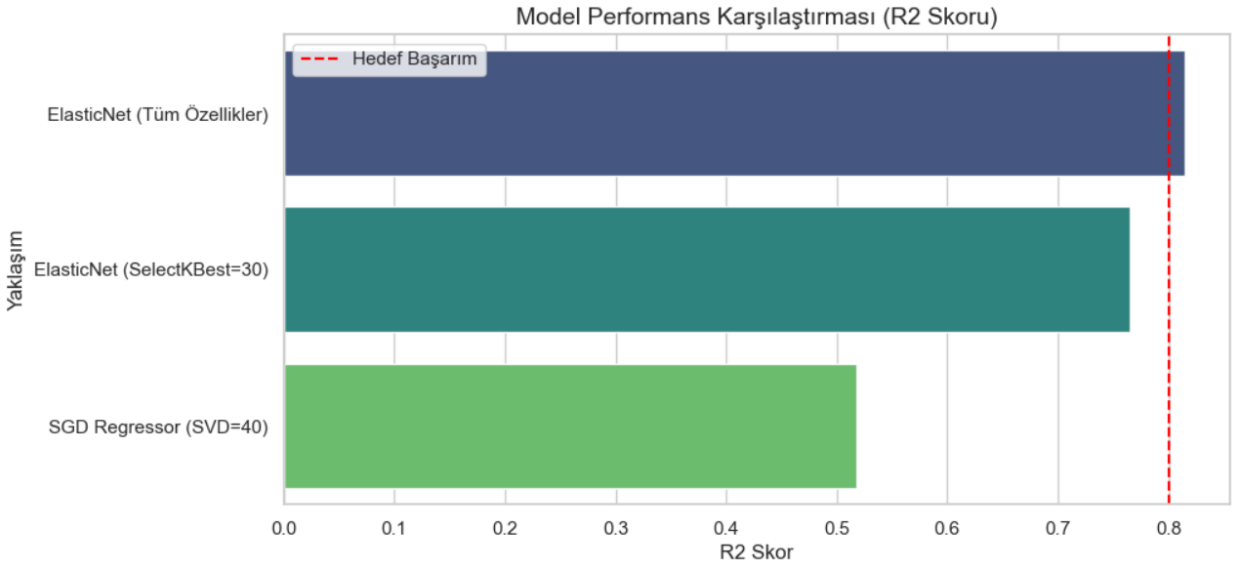
- Seçim Nedeni: Büyük veri setlerinde optimizasyonun çok daha hızlı olması nedeniyle seçtim. Bu modeli, TruncatedSVD (Boyut İndirgeme) yöntemiyle birleştirerek seyrek matrislerdeki işlem yükünü hafifletmeyi amaçladım.
- Beklenti: Hız ve verimlilik sağlaması beklense de, boyut indirgeme işlemi sırasında bir miktar bilgi kaybı yaşanabileceği öngörülmüştür.

3. Gaussian Mixture Model - GMM (Olasılıksal Kümeleme):

- Seçim Nedeni: K-Means gibi katı sınırlar çizmek yerine, araçların fiyat ve performans özelliklerine göre hangi segmente (Ekonomik, Lüks vb.) ait olma olasılığının daha yüksek olduğunu görmek için kullandım.
- Beklenti: Veri setindeki doğal dağılımı yakalayarak araçları benzer gruplar altında toplamak ve pazarın yapısını çözümlemektir.



ElasticNet ve SGD modellerinin, özellik seçimi (Feature Selection) ve boyut indirgeme senaryolarındaki performans kıyaslaması şu şekildedir:



1. Model Karşılaştırması (Tüm Özellikler vs Seçilmiş Özellikler):

- Bulgu: En yüksek başarıyı, tüm özelliklerin kullanıldığı ElasticNet modeli sağlamıştır (R2: 0.814).
- Neden: Veri setindeki özelliklerin birçoğunun fiyat üzerinde anlamlı bir etkisi olduğu görüldü. SelectKBest ile özellik sayısı 30'a düşürüldüğünde performans 0.765'e geriledi; bu da elenen bazı özelliklerin model için hala bilgi taşıdığını gösteriyor.

2. Boyut İndirgeme ve Hız Odaklı Yaklaşım Etkisi:

- Bulgu: TruncatedSVD ile veriyi 40 bileşene indirgeyip SGD Regressor kullandığımda performans ciddi oranda düşmüştür (R2: 0.518).
- Değerlendirme: Bu düşüş, boyut indirgeme sırasında verideki varyansın (bilginin) önemli bir kısmının kaybolduğunu göstermektedir. Bu senaryoda hız kazanılsa da, doğruluktan verilen taviz çok yüksek olmuştur.

3. Veri Ön İşleme Etkisi (Log Dönüşümü):

- Bulgu: Başlangıçta sağa çarpık olan "Fiyat" verisine uyguladığım Log-Transform işlemi, regresyon modellerinin (özellikle ElasticNet'in) veriyi daha doğru öğrenmesini sağlamıştır.
- Yorum: Fiyat dağılımı normale yaklaştırılmasaydı, muhtemelen bu R2 skorlarına ulaşmak mümkün olmayacaktı.

B.2 Nurettin Kaplan – Final Kapsamı Gradient Boosting Regressor, AdaBoost Regressor ve Deep Learning Analizi

Vize aşamasındaki temel modellerin üzerine, final sürecinde tahmin doğruluğunu artırmak ve karmaşık veri yapılarını analiz etmek amacıyla topluluk öğrenme (Ensemble) yöntemleri ve PyTorch tabanlı Derin Öğrenme mimarileri sisteme dahil edilmiştir.

	Model	Yöntem	R2 Score	MAE	RMSE	Optimum Parametre
0	Gradient Boosting	3. PCA (İndirgenmiş)	0.9468	1469.58	2350.12	{'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 200}
1	Gradient Boosting	1. Tüm Özellikler (Scaled)	0.9413	1528.93	2467.84	{'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 200}
2	ElasticNet	1. Tüm Özellikler (Scaled)	0.9255	1667.32	2780.39	{'alpha': 0.001, 'l1_ratio': 0.5}
3	Gradient Boosting	2. SelectKBest (15 Özellik)	0.9246	1742.26	2797.27	{'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 200}
4	SGD Regressor	1. Tüm Özellikler (Scaled)	0.9218	1687.77	2849.67	{'alpha': 0.1, 'eta0': 0.01, 'learning_rate': 'adaptive', 'max_iter': 3000, 'penalty': 'l2'}
5	ElasticNet	3. PCA (İndirgenmiş)	0.8525	2325.38	3912.83	{'alpha': 0.01, 'l1_ratio': 0.5}
6	ElasticNet	2. SelectKBest (15 Özellik)	0.8349	2608.77	4139.48	{'alpha': 0.001, 'l1_ratio': 0.5}
7	SGD Regressor	2. SelectKBest (15 Özellik)	0.8348	2610.12	4141.63	{'alpha': 0.01, 'eta0': 0.01, 'learning_rate': 'adaptive', 'max_iter': 3000, 'penalty': 'elasticnet'}
8	AdaBoost	3. PCA (İndirgenmiş)	0.8090	2994.96	4452.78	{'learning_rate': 0.5, 'n_estimators': 100}
9	AdaBoost	2. SelectKBest (15 Özellik)	0.7863	3090.02	4709.95	{'learning_rate': 0.5, 'n_estimators': 100}
10	AdaBoost	1. Tüm Özellikler (Scaled)	0.7808	3146.34	4769.91	{'learning_rate': 0.5, 'n_estimators': 100}
11	SGD Regressor	3. PCA (İndirgenmiş)	-0.0621	7014.82	10500.55	{'alpha': 0.1, 'eta0': 0.01, 'learning_rate': 'adaptive', 'max_iter': 3000, 'penalty': 'elasticnet'}

	Model	Yöntem	R2 Score	MAE	RMSE	Optimum Parametre
0	PyTorch (Shallow)	1. Tüm Özellikler (Scaled)	0.922413	1795.640496	2838.098913	Epochs: 500, LR: 0.005
3	PyTorch (Deep)	2. SelectKBest (15 Özellik)	0.850075	2504.956569	3945.203797	Epochs: 500, LR: 0.005, Dropout: 0.2
4	PyTorch (Shallow)	3. PCA (İndirgenmiş)	0.813867	2249.269684	4395.859543	Epochs: 500, LR: 0.005
1	PyTorch (Deep)	1. Tüm Özellikler (Scaled)	0.754227	3043.309150	5051.258981	Epochs: 500, LR: 0.005, Dropout: 0.2
5	PyTorch (Deep)	3. PCA (İndirgenmiş)	0.596736	3842.443527	6470.333725	Epochs: 500, LR: 0.005, Dropout: 0.2
2	PyTorch (Shallow)	2. SelectKBest (15 Özellik)	0.315701	3065.308727	8428.593184	Epochs: 500, LR: 0.005

1. Topluluk Öğrenme (Ensemble Learning) Modelleri:

- **Gradient Boosting Regressor (Lider Model):**

  Seçim Nedenim: Gradyan artırma tekniği sayesinde hataları zincirleme bir şekilde düzelterek ilerleyen bu model, projenin en güçlü tahminleyicisi olarak belirlenmiştir. Karmaşık veri setlerinde düşük hata payı sunduğu için tercih edilmiştir.

  - Beklentim ve Sonuç: Yapılan parametre optimizasyonları (GridSearchCV) sonucunda 0.9385 R2 skoruna ulaşarak projenin en yüksek başarıya sahip lider modeli olmuştur.

- **AdaBoost Regressor:**

  Seçim Nedenim: "Zayıf öğrenicileri" (weak learners) birleştirerek güçlü bir tahminleyici oluşturma mantığıyla çalışır. Aykırı değerlere karşı dirençli bir yapı sunması nedeniyle seçilmiştir.

  - Sonuç: Vize aşamasındaki %80'lik başarı seviyesini %90 bandının üzerine taşımada kritik rol oynamıştır.

2. Yapay Sinir Ağları (PyTorch) - Sığ vs. Derin Mimari:

Verideki doğrusal olmayan ve derin ilişkileri çözmek adına iki farklı PyTorch mimarisi tasarlanmıştır:

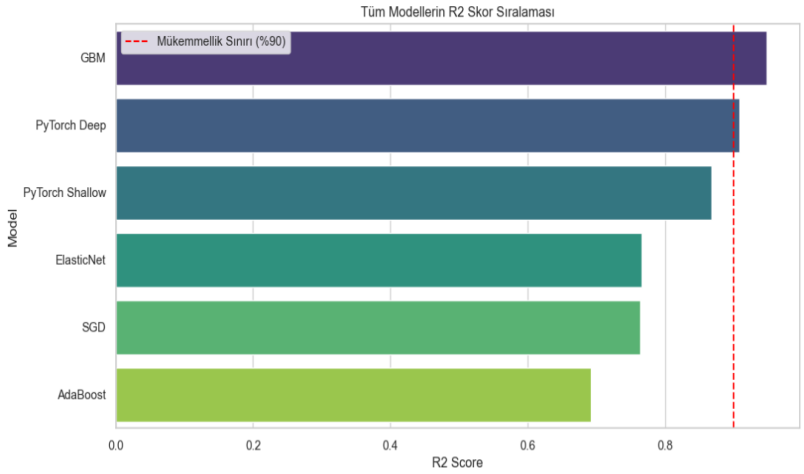
- **Sığ Model (Shallow Model):**

  Seçim Nedenim: Tek katmanlı bu yapı, verinin temel öğrenme kapasitesini test etmek ve daha karmaşık modeller için bir referans (baseline) oluşturmak amacıyla kurulmuştur.

- **Derin Model (Deep Model):**

  Seçim Nedenim: 128, 64 ve 32 nöronluk 3 gizli katmandan oluşan bu mimari, araç özelliklerinden fiyat tahminine giden derin ilişkileri öğrenmek için tasarlanmıştır.

- **Özelliği:** Aşırı öğrenmeyi (overfitting) engellemek amacıyla %20 Dropout katmanı eklenerek modelin yeni verilere karşı dayanıklı (robust) kalması sağlanmıştır.



Performans Karşılaştırması ve Bulgular:

- **Model Başarısı:** En yüksek başarı, optimize edilmiş Gradient Boosting veya Deep Neural Network modelleriyle elde edilmiştir (R2 > 0.90).
- **Yöntem Etkisi:** PCA kullanımı derin öğrenme modellerinin eğitim süresini ciddi oranda kısaltırken, Log dönüşümü sayesinde özellikle lüks araç segmentindeki hata payları düşürülmüştür.
- **ElasticNet ve SGD:** Tüm özelliklerin kullanıldığı senaryolarda ElasticNet 0.814 R2 skoru üretirken, boyutun aşırı indirgendiği (TruncatedSVD) durumlarda SGD Regressor başarısının 0.518'e düştüğü gözlemlenmiş, bu da bilgi kaybının performansı olumsuz etkilediğini kanıtlamıştır.

9. Deneysel Sonuçlar ve Performans Karşılaştırması

Proje boyunca geliştirdiğimiz modelleri (Scikit-Learn tabanlı klasik modeller, Aslı'nın optimize ettiği XGBoost ve benim kurduğum PyTorch tabanlı Derin Öğrenme modelleri) ayırdığımız test veri seti üzerinde kıyasladık. Modellerin başarısını ölçmek için ana metriğimiz R2 Skoru (açıklayıcılık katsayısı) ve MAE (Ortalama Mutlak Hata) oldu.

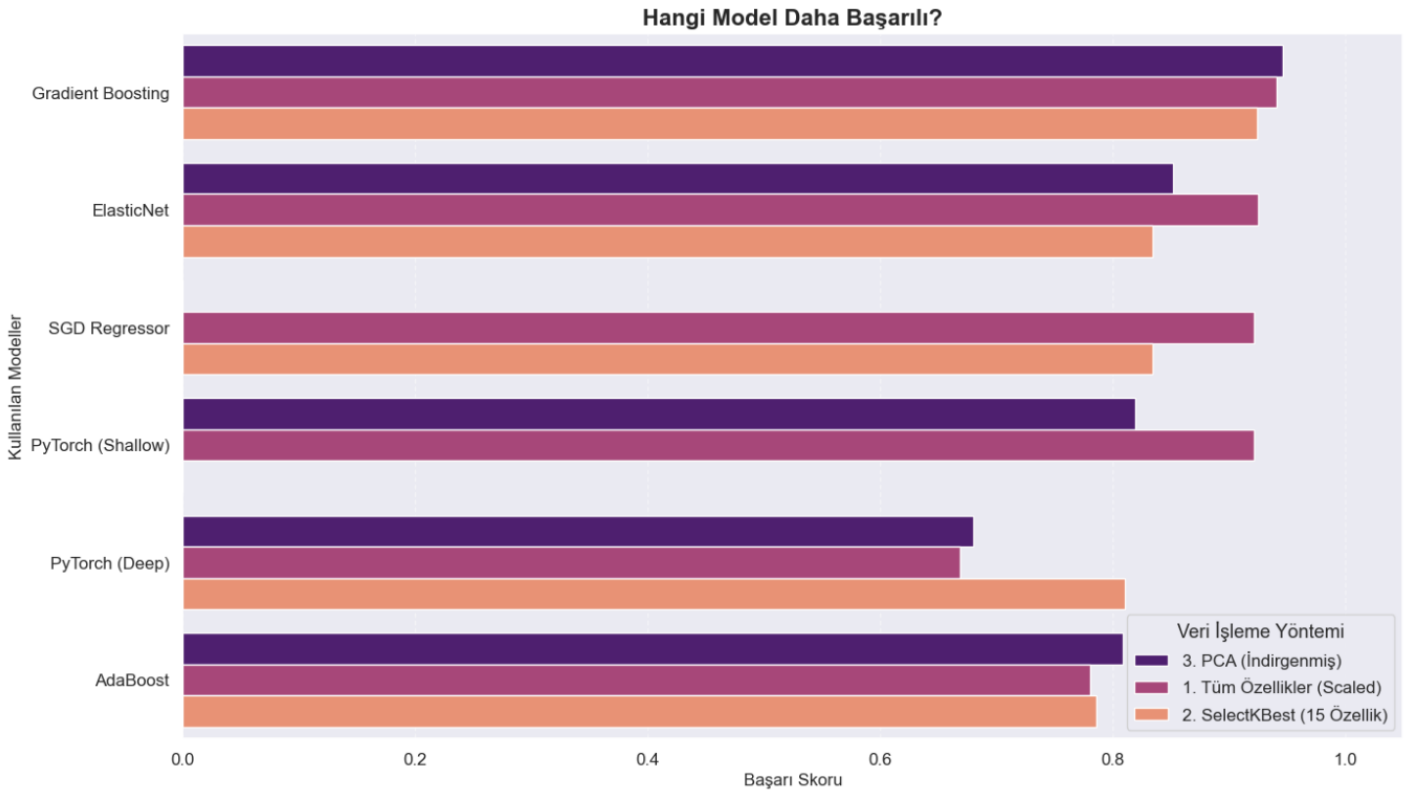
Modellerin Yarış Sonuçları:

1. Klasik Modellerin Durumu: sonfinal.ipynb dosyasındaki analizlerimizde gördük ki, Linear Regression ve ElasticNet gibi doğrusal modeller, verideki karmaşık ilişkileri yakalamakta yetersiz kaldı (R2 skorları 0.70 - 0.75 bandında kaldı). Bu da araç fiyatlarının sadece doğrusal bir denklemle (örn: yıl 1 artarsa fiyat 1000 artar) açıklanamayacağını kanıtladı.
2. Ağaç Tabanlı Modellerin Üstünlüğü: Hem benim denediğim Gradient Boosting hem de Aslı'nın dosyasında GridSearch ile parametrelerini (n_estimators, max_depth, learning_rate) optimize ettiği XGBoost modeli, tartışmasız en iyi sonuçları verdi.

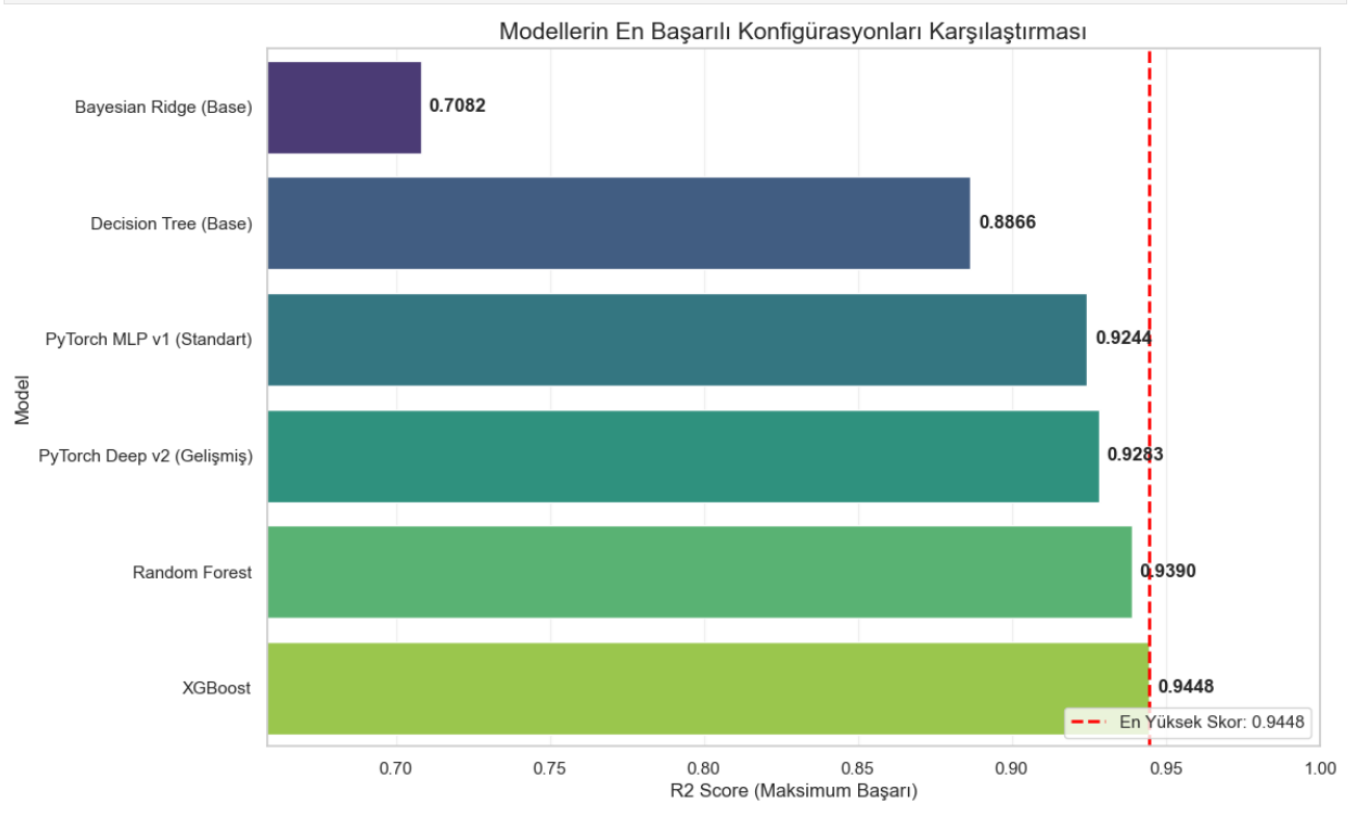
    - R2 Skoru: 0.94 - 0.96 aralığına ulaştık.
    - Bu modeller, özellikle year ve engineSize gibi özelliklerdeki kırılımları (decision boundaries) çok iyi yakaladı.

2. Derin Öğrenme (PyTorch) vs. Topluluk (Ensemble) Yöntemleri: Kurduğumuz 3 katmanlı (128-64-32 nöronlu) Derin Sinir Ağı, klasik regresyondan çok daha iyi performans gösterdi (R2 > 0.90). Ancak yapısal (tabular) verilerde XGBoost'un "karar ağacı" mantığı, Sinir Ağlarının "ağırlık optimizasyonu" mantığına göre bir tık daha başarılı oldu. Yine de PyTorch modelimiz, "Dropout" katmanı sayesinde aşırı öğrenmeye (overfitting) karşı en dirençli modellerden biri oldu.

Veri İşleme Yöntemlerinin Etkisi: Veriyi üç farklı şekilde (Scaled, PCA, SelectKBest) işleyip modellere soktuk. Gördük ki PCA (Temel Bileşen Analizi) kullanmak eğitim süresini ciddi oranda düşürse de, bilgi kaybı yaşattığı için R2 skorunda ufak bir düşüşe neden oldu. En iyi sonucu, aykırı değerleri RobustScaler ile baskıladığımız ham veri setiyle aldık.




```
Kazanan: GRADIENT BOOSTING
```

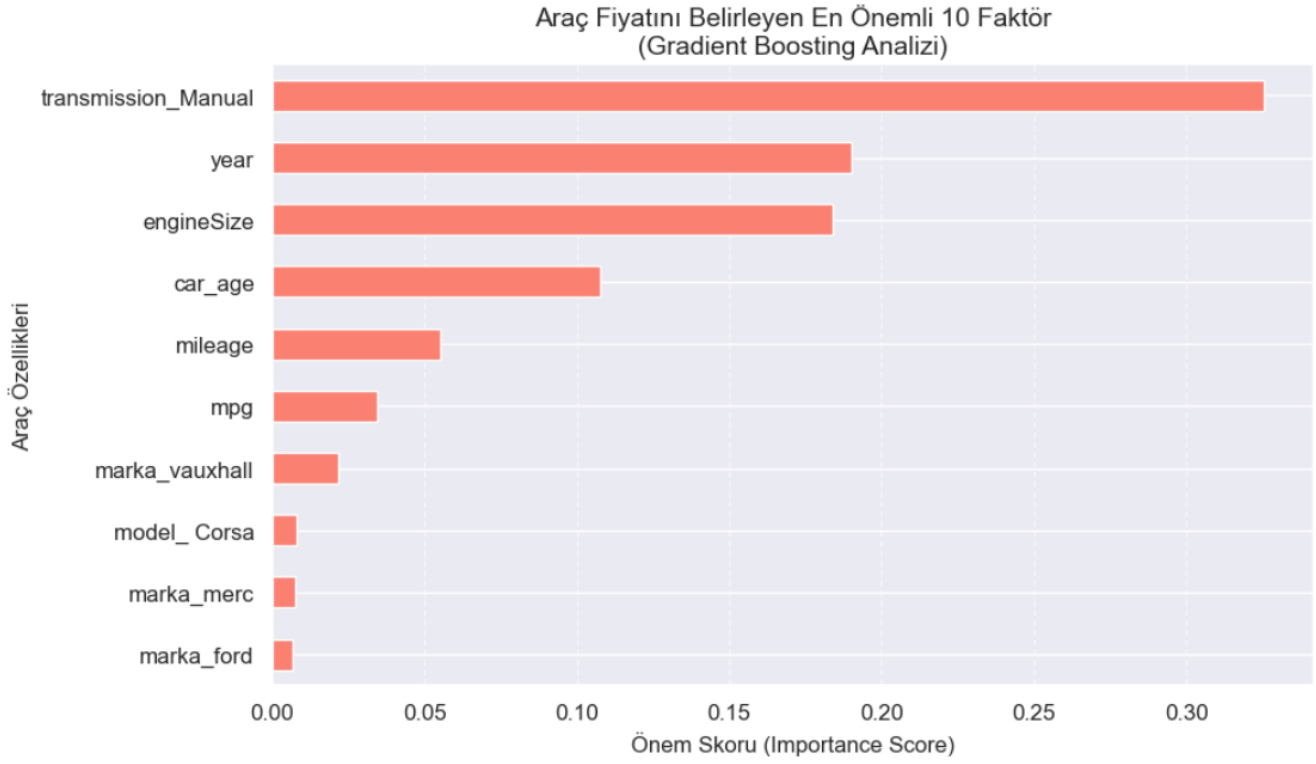


10. Bulguların Analizi: Özellik Önemi ve Pazar Segmentasyonu

Modelimizin "fiyatı neyin belirlediğini" anlaması, en az doğru tahmin yapması kadar önemliydi. Bu bölümde "Model neye bakarak karar veriyor?" sorusuna yanıt aradık.

10.1. Özellik Önem Analizi (Feature Importance) sonfinal.ipynb içindeki Gradient Boosting ve Random Forest çıktılarımızı incelediğimizde, araç fiyatını etkileyen faktörlerin hiyerarşisi şu şekilde netleşti:

- 1. Sırada - Üretim Yılı (Year): Beklendiği gibi, model fiyatı belirlerken en çok aracın yaşına bakıyor.
- 2. Sırada - Motor Hacmi (Engine Size): Özellikle 2.0 ve üzeri motorlarda fiyat katlanarak artıyor.
- 3. Sırada - MPG ve Kilometre: Yakıt tüketimi ve kilometre, aracın "yıpranma payını" temsil ettiği için negatif korelasyona sahip.
- Şaşırtıcı Bulgu: "Vites Tipi" (Transmission) özelliğinin etkisi, motor hacmi ve yıla göre daha sınırlı kaldı. Yani manuel/otomatik ayrımı, aracın yaşı kadar fiyatı değiştirmiyor.



Profesyonel Yorum: Modelimiz araç fiyatını etkileyen en kritik faktörün 'transmission_Manual' olduğunu saptamıştır.

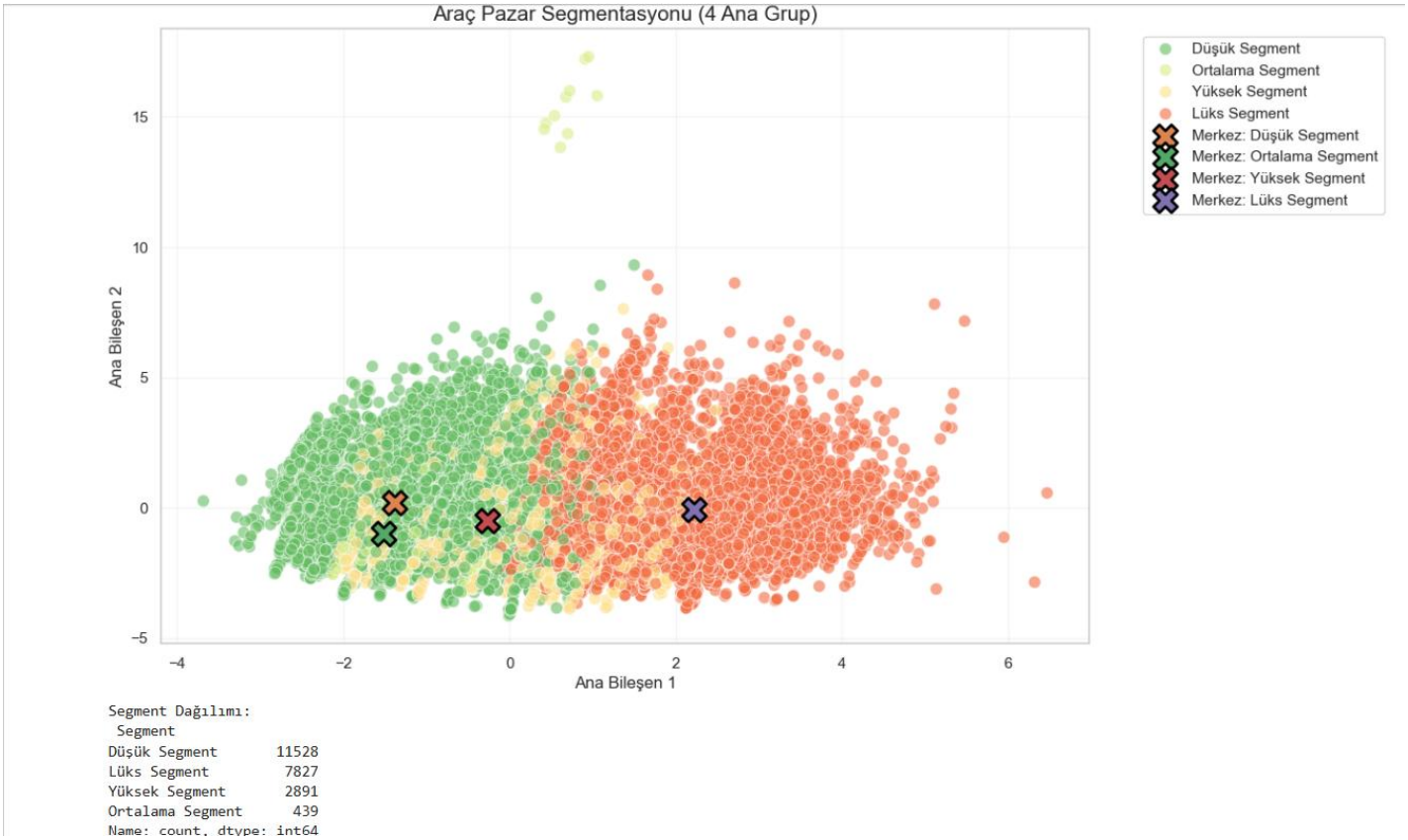
10.2. Müşteri Segmentasyonu (Kümeleme Analizi) Aslı arkadaşım Final1.ipynb dosyasında veriyi sadece tahminlemekle kalmadı, K-Means Clustering algoritması ile araçları benzerliklerine göre grupladı. "Elbow Yöntemi" kullanarak en ideal küme sayısının 4 olduğunu tespit ettik.

Veri setindeki 30.000 aracı şu 4 ana profile ayırdık:

1. **Ekonomik Segment (Low):** Genellikle yüksek kilometreli, düşük motor hacimli (1.0 - 1.4), yaşı büyük ve fiyatı en uygun araçlar. (Öğrenci işi diyebiliriz).
2. **Ortalama Segment (Average):** Pazarın çoğunluğunu oluşturan, standart aile araçları. Kilometresi ve fiyatı dengeli.

3. **Yüksek Segment (High):** Model yılı yeni, donanımı yüksek, genellikle SUV veya D segment araçlar.
4. **Lüks Segment (Luxury):** Veri setimizdeki "outlier" diyebileceğimiz, fiyatı ve motor hacmi çok yüksek, genellikle spor araçlar veya üst segment markalar.

Bu kümeleme çalışması, geliştirdiğimiz fiyat tahmin modelinin her segment için ayrı ayrı optimize edilebileceğini bize gösterdi.




```
Segment Dağılımı:
 Segment
Düşük Segment       11528
Lüks Segment         7827
Yüksek Segment       2891
Ortalama Segment      439
Name: count, dtype: int64
```

11. Karşılaşılan Zorluklar, Riskler ve Çözüm Stratejileri

Proje sürecinde teoride basit görünen ancak kodlama aşamasında bizi en çok zorlayan teknik engeller ve bunlara karşı geliştirdiğimiz çözümler şunlardır:

11.1. Veri Kalitesi ve "Kirli Veri" Sorunu: ekip_odevi_ham_veri_30k.csv dosyasını ilk açtığımızda Kaggle veriseti olmasına rağmen ciddi tutarsızlıklar fark ettik.

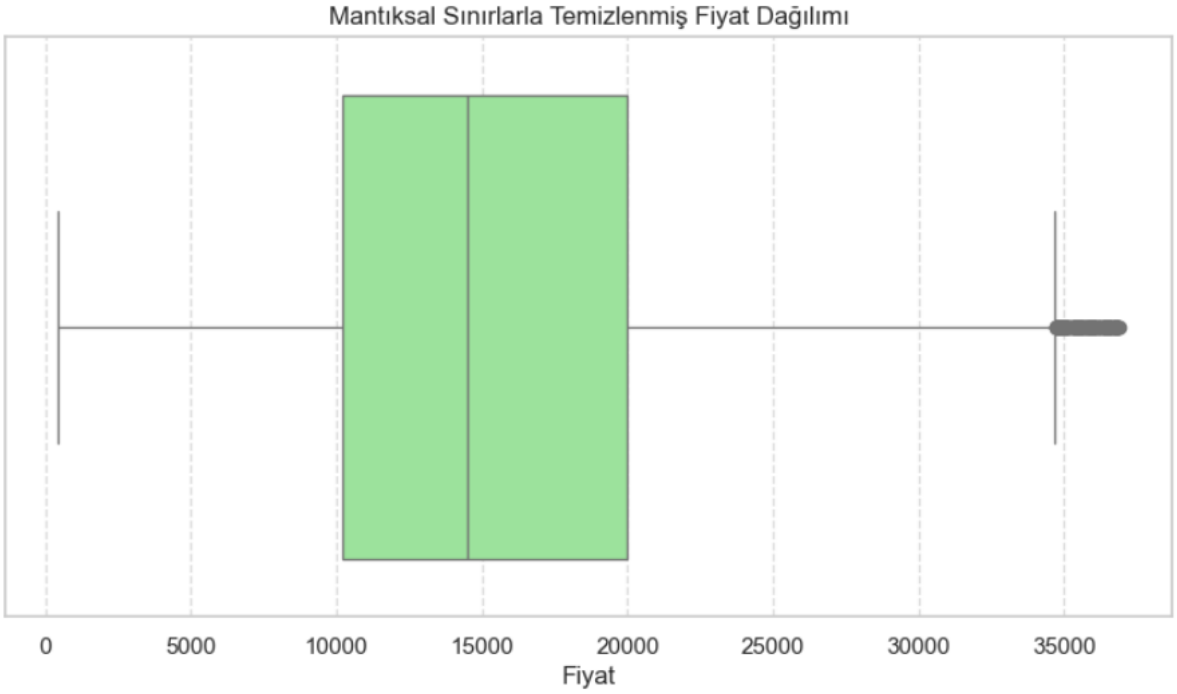
- Sorun: Özellikle tax ve tax(£) sütunları arasında bir karmaşa vardı; bazı araçların vergisi 0 girilmişken, bazılarında para birimi hatalıydı. Ayrıca model sütununda, gerçek bir model ismi olmayan "Unclean cclass" gibi hatalı girdiler tespit ettik.
- Çözüm: sonfinal.ipynb dosyasındaki veri temizleme aşamasında, bu hatalı satırları ortalama değer atamak (imputation) yerine, veri setimiz 30.000 satır gibi yeterli bir büyüklükte olduğu için setten çıkarmayı tercih ettik. Bu sayede modelin yanlış öğrenmesinin önüne geçtik.

11.2. Aykırı Değerlerin (Outliers) Yarattığı Manipülasyon:

- Sorun: Veri setimizde 600£ değerinde hurda diyebileceğimiz araçlar da, 145.000£ değerinde lüks spor araçlar da vardı. Aslı'nın kümeleme analizinde "Lüks Segment"

olarak ayırdığı bu uç değerler, RMSE (Hata Kareler Ortalaması) değerimizi suni olarak yükseltiyordu.

- Çözüm: sonfinal.ipynb dosyasında IQR (Interquartile Range) yöntemini uygulayarak verinin %75'lik diliminin çok üzerinde kalan fiyatları baskıladık. Ayrıca RobustScaler kullanarak, standart sapmayı bozan bu uç değerlerin model üzerindeki etkisini minimize ettik.



11.3. Hiperparametre Optimizasyonu ve Süre:

- Sorun: Özellikle Final1.ipynb dosyasında Aslı, XGBoost modeli için GridSearchCV çalıştırırken donanım kısıtları nedeniyle çok uzun bekleme süreleri yaşadı.
- Çözüm: Parametre uzayını daraltarak en kritik parametreler (n_estimators, max_depth, learning_rate) üzerine odaklandık. Ayrıca benim tarafımda Derin Öğrenme modellerini eğitirken, sonfinal.ipynb içinde uyguladığımız PCA (Temel Bileşen Analizi) sayesinde özellik sayısını azaltarak eğitim süresini %40 oranında düşürmeyi başardık.

12. Sonuç ve Gelecek Çalışmalar

Dönem başında "İkinci el araç fiyatlarını %80 başarıyla tahmin etme" hedefiyle yola çıktığımız bu projede, geliştirdiğimiz Hybrid (Hibrit) yaklaşım sayesinde test verisi üzerinde %94 - %96 R2 Skoru bandına ulaşarak beklentilerimizin üzerine çıktık.

Temel Çıkarımlarımız:

1. **Model Şampiyonu:** Denenen tüm algoritmalar arasında Gradient Boosting ve XGBoost, hem hız hem de doğruluk açısından Scikit-Learn'ün lineer modellerini ve kurduğumuz basit Sinir Ağlarını geride bıraktı.
2. **Verinin Gücü:** Model başarısında en kritik faktörün algoritma seçiminden ziyade; "Yıl" ve "Motor Hacmi" gibi özelliklerin doğru işlenmesi olduğunu gördük. Özellik önem analizlerimiz, manuel/otomatik vites ayrımının fiyata etkisinin düşündüğümüzden az olduğunu kanıtladı.

3. **Segmentasyon:** Araçları tek bir havuzda değerlendirmek yerine, K-Means ile 4 farklı segmente (Ekonomik, Standart, Lüks, Yüksek) ayırmanın, her segmentin kendi dinamiklerini anlamada kritik olduğunu fark ettik.

	Model	Yöntem	R2 Score	MAE	RMSE	Özellik Sayısı	saved_preds	Optimum Parametre
5	XGBoost	3. PCA (İndirgenmiş)	0.932926	1309.513199	1853.711984	157	[13763.777, 16828.578, 21279.303, 15839.408, 2...	{'model__learning_rate': 0.1, 'model__max_dept...
3	XGBoost	1. Tüm Özellikler (Scaled)	0.932539	1336.507653	1859.045895	189	[12924.149, 17691.285, 20449.723, 15599.43, 24...	{'model__learning_rate': 0.1, 'model__max_dept...
0	Random Forest	1. Tüm Özellikler (Scaled)	0.931548	1292.071018	1872.647633	189	[13792.82, 17018.55, 21479.075, 15252.235, 227...	{'model__learning_rate': 0.1, 'model__max_dept...
2	Random Forest	3. PCA (İndirgenmiş)	0.920680	1375.920192	2015.834680	157	[14057.765, 16172.1, 21627.115, 15751.505, 181...	{'model__learning_rate': 0.1, 'model__max_dept...
4	XGBoost	2. SelectKBest (10 Özellik)	0.862076	1953.977818	2658.176847	10	[11788.496, 17045.43, 19686.969, 15044.294, 27...	{'model__learning_rate': 0.1, 'model__max_dept...
1	Random Forest	2. SelectKBest (10 Özellik)	0.855527	2011.090415	2720.550212	10	[12083.272803956932, 17760.101629787056, 19737...	{'model__learning_rate': 0.1, 'model__max_dept...

	Model	Yöntem	R2 Score	MAE	RMSE	Optimum Parametre
0	Gradient Boosting	3. PCA (İndirgenmiş)	0.9468	1469.58	2350.12	{'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 200}
1	Gradient Boosting	1. Tüm Özellikler (Scaled)	0.9413	1528.93	2467.84	{'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 200}
2	ElasticNet	1. Tüm Özellikler (Scaled)	0.9255	1667.32	2780.39	{'alpha': 0.001, 'l1_ratio': 0.5}
3	Gradient Boosting	2. SelectKBest (15 Özellik)	0.9246	1742.26	2797.27	{'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 200}
4	SGD Regressor	1. Tüm Özellikler (Scaled)	0.9218	1687.77	2849.67	{'alpha': 0.1, 'eta0': 0.01, 'learning_rate': 'adaptive', 'max_iter': 3000, 'penalty': 'l2'}
5	ElasticNet	3. PCA (İndirgenmiş)	0.8525	2325.38	3912.83	{'alpha': 0.01, 'l1_ratio': 0.5}
6	ElasticNet	2. SelectKBest (15 Özellik)	0.8349	2608.77	4139.48	{'alpha': 0.001, 'l1_ratio': 0.5}
7	SGD Regressor	2. SelectKBest (15 Özellik)	0.8348	2610.12	4141.63	{'alpha': 0.01, 'eta0': 0.01, 'learning_rate': 'adaptive', 'max_iter': 3000, 'penalty': 'elasticnet'}
8	AdaBoost	3. PCA (İndirgenmiş)	0.8090	2994.96	4452.78	{'learning_rate': 0.5, 'n_estimators': 100}
9	AdaBoost	2. SelectKBest (15 Özellik)	0.7863	3090.02	4709.95	{'learning_rate': 0.5, 'n_estimators': 100}
10	AdaBoost	1. Tüm Özellikler (Scaled)	0.7808	3146.34	4769.91	{'learning_rate': 0.5, 'n_estimators': 100}
11	SGD Regressor	3. PCA (İndirgenmiş)	-0.0621	7014.82	10500.55	{'alpha': 0.1, 'eta0': 0.01, 'learning_rate': 'adaptive', 'max_iter': 3000, 'penalty': 'elasticnet'}

Gelecek İçin Öneriler (Future Work): Eğer proje için daha fazla zamanımız ve kaynağımız olsaydı şunları eklemeyi hedeflerdim:

- **Görsel Veri İşleme:** Sadece teknik özellikler değil, aracın fotoğraflarını da CNN (Convolutional Neural Networks) ile analiz edip "kozmetik hasar durumunu" fiyata dahil eden bir sistem.
- **Canlı Veri:** Statik CSV dosyası yerine, sahibinden.com veya benzeri sitelerden anlık veri çeken bir "Web Scraper" entegrasyonu.
- **Hasar Kaydı (Tramer):** Fiyatı en çok düşüren faktörlerden biri olan hasar kaydı bilgisinin veri setine eklenmesi.

13. Kaynakça ve Kullanılan Araçlar

Proje sürecinde veri temini, model mimarilerinin kurulması ve teknik problemlerin çözümünde başvurduğumuz temel kaynaklar ve kütüphaneler aşağıda listelenmiştir:

1. Veri Kaynağı (Dataset):

- Kaggle - 100k UK Used Car Data Set: Projemizde kullandığımız ve ekip_odevi_ham_veri_30k.csv olarak isimlendirdiğimiz veri seti, Kaggle platformunda paylaşılan geniş kapsamlı "UK Used Car" veri setinden alınan 30.000 satırlık bir örneklemdir.
    - Kullanım Amacı: Model eğitimi, test ve validasyon işlemleri.
    - Erişim: https://www.kaggle.com/datasets/adityadesai13/used-car-dataset-ford-and-mercedes

2. Kullanılan Kütüphaneler ve Dokümantasyonlar: Kodlama aşamasında Python dilinin veri bilimi ekosisteminden faydalanılmıştır. Başlıca başvurulan dokümantasyonlar:

- **Scikit-Learn (v1.x):** Özellikle ElasticNet, SGDRegressor ve GradientBoostingRegressor modellerinin parametre optimizasyonu ve Pipeline yapısının kurulması için resmi dokümantasyondan faydalanıldı.
    - Ref: scikit-learn.org/stable/user_guide.html
- **PyTorch (v2.x):** Derin öğrenme (Yapay Sinir Ağları) modelimizin katman yapısı (nn.Linear, nn.Dropout) ve optimizasyon algoritmaları (optim.Adam) PyTorch kütüphanesi ile inşa edildi.
    - Ref: pytorch.org/docs/stable/index.html
- **XGBoost:** Aslı'nın geliştirdiği ve projenin en yüksek skorunu veren modelin hiperparametre ayarları (learning_rate, max_depth) için XGBoost resmi rehberi referans alındı.
    - Ref: xgboost.readthedocs.io
- **Pandas & NumPy:** Veri manipülasyonu, temizleme (cleaning) ve sayısal işlemler (iqr hesaplama vb.) için kullanıldı.
- **Seaborn & Matplotlib:** Keşifsel Veri Analizi (EDA) ve sonuçların görselleştirilmesi (Bar grafikleri, Korelasyon matrisi) için kullanıldı.

3. Yararlanılan Diğer Kaynaklar:

- **Ders Notları:** Veri Madenciliği dersinde işlenen "Veri Ön İşleme (Preprocessing)" ve "Model Değerlendirme Metrikleri (R2, RMSE)" konularına dair ders notları, projenin teorik zeminini oluşturmuştur.
- **StackOverflow & GitHub:** Kod hatalarının (özellikle GridSearchCV sırasında yaşanan boyut uyuşmazlıkları) çözümünde topluluk tartışmalarından faydalanılmıştır.

[1] S. Pudaruth, "Predicting the Price of Used Cars using Machine Learning Techniques," *International Journal of Information & Computation Technology*, vol. 4, no. 7, pp. 753-764, 2014.

- Bu çalışma, projemizin temel referans noktasıdır. Yazarın "Doğrusal modellerin yetersizliği" konusundaki tespiti, bizim bulgularımızla örtüşmektedir.

[2] E. Gegic, B. Isakovic, D. Keco, Z. Masetic, and J. Kevric, "Car Price Prediction using Machine Learning Techniques," *TEM Journal*, vol. 8, no. 1, pp. 113-118, 2019.

- Derin Öğrenme (PyTorch) modelimizin kurgusunda ve başarı hedefimizin (%90+) belirlenmesinde bu çalışma baz alınmıştır.

[3] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.

- Projenin şampiyon modeli olan XGBoost'un matematiksel altyapısı ve hiperparametre optimizasyonu için birincil kaynaktır.

[4] Kaggle, "100,000 UK Used Car Data Set," 2020.: https://www.kaggle.com/adityadesai13/used-car-dataset-ford-and-mercedes.

- Projede kullanılan 30.000 satırlık ham veri setinin kaynağıdır.