

R Regresyon Final

Kaan, Sadık, Alihan

2026-01-27

BMW İkinci El Otomobil Değer Analizi

Bu çalışma kapsamında, Türkiye ikinci el otomobil piyasasındaki **BMW** marka araçların fiyatlandırma dinamiklerini anlamak amacıyla oluşturulan **10.000** gözlemlilik sentetik bir veri seti incelenmiştir. Veri seti; araçların teknik konfigürasyonlarını, kullanım geçmişlerini ve detaylı ekspertiz (hasar) kayıtlarını içermektedir.

Veri Sözlüğü

Aşağıdaki tablo, analizde kullanılan değişkenlerin teknik detaylarını ve açıklamalarını içermektedir:

1. Teknik ve Kimlik Bilgileri

Değişken	Tanım	Birim / Format
series	Aracın Model Serisi	Factor (1 Serisi, 3 Serisi, X Serisi...)
segment	Aracın Pazar Segmenti	Factor (C, D, E...)
horsepower	Motorun Ürettığı Güç	Beygir Gücü (HP)
curb_weight	Aracın Boş Ağırlığı	Kilogram (kg)
top_speed	Maksimum Sürat	km/saat

2. Kullanım Geçmişi

Değişken	Tanım	Birim / Format
model_year	Aracın Üretim Yılı	Yıl (YYYY)
mileage	Aracın Toplam Kat Ettiği Mesafe	Kilometre (km)

3. Ekspertiz ve Hasar Kayıtları

Değişken	Tanım	Birim / Format
roof_replaced	Tavan Değişim Durumu	Binary (1: Evet, 0: Hayır)
roof_painted	Tavan Tam Boya Durumu	Binary (1: Evet, 0: Hayır)
hood_replaced	Kaput Değişim Durumu	Binary (1: Evet, 0: Hayır)
trunk_replaced	Bagaj Kapağı Değişim Durumu	Binary (1: Evet, 0: Hayır)

Değişken	Tanım	Birim / Format
door_..._count	Kapılardaki Değişen/Boyalı Adedi	Tam Sayı (Numeric)
fender_..._count	Çamurluklardaki Değişen/Boyalı Adedi	Tam Sayı (Numeric)
bumper_process	Tamponda Yapılan İşlem Sayısı	Tam Sayı (Numeric)
is_heavy_damaged	Ağır Hasar (Pert) Kaydı Durumu	Factor (1: Evet, 0: Hayır)

4. Hedef Değişken

Değişken	Tanım	Birim / Format
price	Aracın Piyasa Satış Fiyatı	Türk Lirası (TL)

Analitik Yaklaşım

Veri seti üzerinde gerçekleştirilecek analizler şu temel direklere dayanmaktadır:

- **Veri Ön İşleme:** Eksik verilerin (Missing Values) serisi özel **Mod** ve **MICE** algoritmaları ile doldurulması.
- **Özellik Mühendisliği (Feature Engineering):** Parça bazlı hasarların kümülatif bir **Hasar Skoru** (Damage Score) altında birləşdirilmesi.
- **Tahminleme:** Araç fiyatı üzerindeki doğrusal olmayan (non-linear) etkilerin logaritmik regresyon modelleri ile incelenmesi.

```
bmw_cars <- read.csv("bmw_data.csv")
```

Değişen / Boyanmış / Local Boyanmış Parçalar için Hasar Skoru (Boyut İndirgeme)

1. Parçaların Önem Sırası (Hierarchy)

Her parça araç değerini aynı oranda düşürmez. **Skor yaklaşımı**, modele “Tavan değişiminin (puan) bir kapı boyasından (puan) çok daha kritik” olduğunu öğretmemizi sağlar.

2. Veri Yoğunluğunu Artırmak (Denser Signal)

16 farklı kolonda tek tük bulunan hasar kayıtları “dağınık” bir gürültü yaratır. Tüm bunları tek bir **risk_score** altında topladığımızda, modelin eline hasarın şiddetini gösteren çok daha net ve “anlamlı” bir veri vermiş oluruz.

3. Yorumlama Kolaylığı (Interpretability)

16 farklı değişkenin katsayısını tek tek açıklamak yerine, “Hasar skoru 1 birim arttığında fiyat %X düşüyor” diyerek çok daha profesyonel ve anlaşılır bir sonuç elde ederiz.

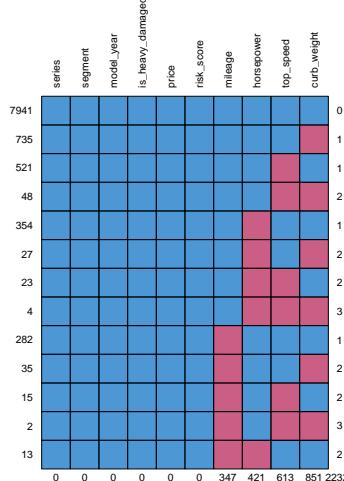
Özellik	Neden Skor Yaklaşımı?
Mantık	Tavan > Kaput > Kapı hiyerarşisini kurmak.
İstatistik	Dağınık veriyi tek bir güdü sinyale dönüştürmek.
Sonuç	Daha basit ve yorumlanabilir bir model elde etmek.

```
bmw_cars <- bmw_cars %>%
  mutate(risk_score =
    (roof_replaced * 150) + (roof_painted * 75) + (roof_local_painted * 40) +
    (hood_replaced * 60) + (hood_painted * 30) + (hood_local_painted * 15) +
    (trunk_replaced * 40) + (trunk_painted * 20) + (trunk_local_painted * 10) +
    (door_replaced_count * 10) + (door_painted_count * 5)
    + (door_local_painted_count * 2) + (fender_replaced_count * 8) + (fender_painted_count * 4) +
    (bumper_process_count * 1)
  ) %>% select(-roof_replaced, -roof_painted, -roof_local_painted,
               -hood_replaced, -hood_painted, -hood_local_painted,
               -trunk_replaced, -trunk_painted, -trunk_local_painted,
               -door_replaced_count, -door_painted_count, -door_local_painted_count,
               -fender_replaced_count, -fender_painted_count, -fender_local_painted_count,
               -bumper_process_count)
```

Boş Gözlem Doldurma

1. MICE paketi ile boş gözlemlere bakmak

```
invisible(md.pattern(bmw_cars,
                     rotate.names = TRUE,
                     plot = TRUE))
```



2. Seri Bazlı Segment Doldurma

- **Model Karakteristiğini Korumak:** BMW serileri (1, 3, 5, X Serisi vb.) arasında beygir gücü, ağırlık ve fiyat gibi değişkenler açısından belirgin hiyerarşik farklar bulunur.
- **Veri Yanlılığını (Bias) Önlemek:** * **Genel Atama:** Tüm veri setinin medyanını kullanmak, 1 Serisi (küçük/hafif) ile X5 (büyük/ağır) araçları birbirine yaklaştırarak veriyi “ortalama bir modele” hapseder.
- **Seriye Özel Atama:** 5 Serisi’ndeki eksik bir teknik verinin, yine diğer 5 Serisi araçların ortalamasıyla doldurulması, aracın fabrikasyon kimliğine en yakın sonucun üretilmesini sağlar.

```
bmw_cars <- bmw_cars %>%
  mutate(segment = na_if(segment, ""))
  group_by(series) %>%
  mutate(
    segment = if_else(
      is.na(segment),
      {
        seg <- segment[!is.na(segment)]
        if (length(seg) == 0) NA_character_
        else names(which.max(table(seg)))
      },
      segment
    )
  ) %>%
  ungroup()
```

3. MICE ve lm() için değişkenleri faktöre çevirme

```
bmw_cars$series <- as.factor(bmw_cars$series)
bmw_cars$segment <- as.factor(bmw_cars$segment)
bmw_cars$is_heavy_damaged <- as.factor(bmw_cars$is_heavy_damaged)
```

4. MICE ile Random Forest ile boş gözlem doldurma

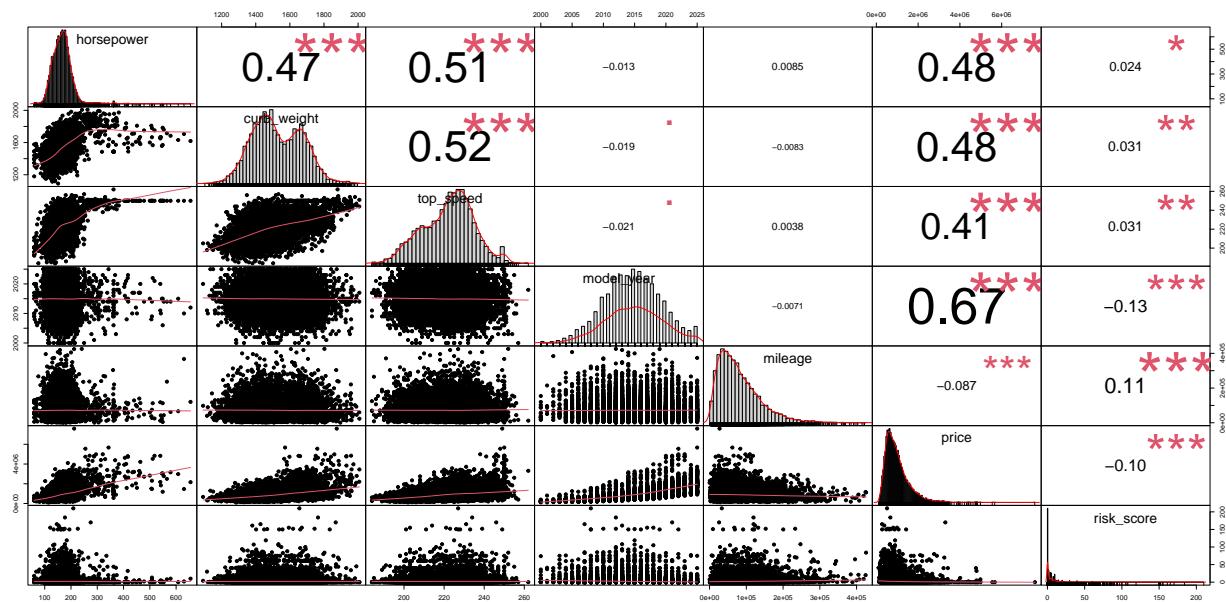
```
set.seed(301)
imputed = mice(bmw_cars, method = "rf", m = 3, printFlag = F)
data_Imp = complete(imputed, 1)
```

Train-test ayrimı

```
set.seed(301)
sampleIndex <- sample(1:nrow(data_Imp), size = 0.8 * nrow(data_Imp))

trainset <- data_Imp[sampleIndex,]
testset <- data_Imp[-sampleIndex,]
```

Korelasyon grafiğinin incelenmesi



Modelin Kurulması

1. Hiçbir işlem olmadan kurulan model

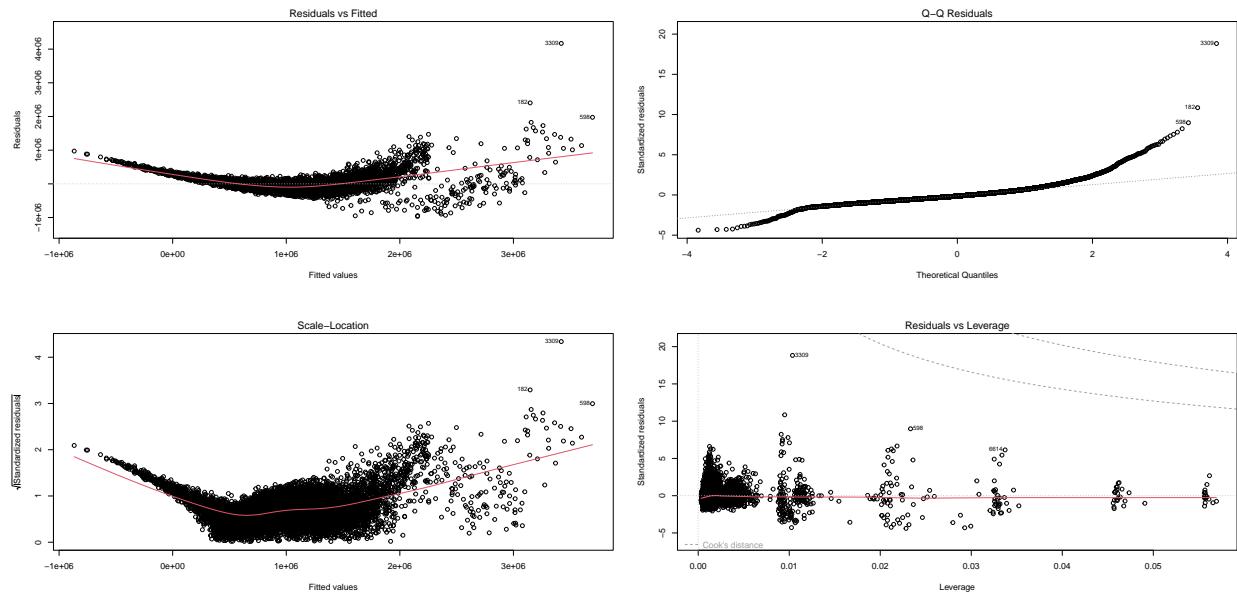
```
model_without_log <- lm(price ~ ., data = trainset)
summary(model_without_log)
```

```
##
## Call:
## lm(formula = price ~ ., data = trainset)
##
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -965371 -124495  -30918   82332  4169731 
##
## Coefficients: (3 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -1.747e+08  1.104e+06 -158.226 < 2e-16 ***
## series2 Serisi  6.178e+04  1.435e+04   4.305 1.69e-05 ***
## series3 Serisi  3.367e+05  1.063e+04   31.663 < 2e-16 ***
## series4 Serisi  5.161e+05  1.717e+04   30.061 < 2e-16 ***
## series5 Serisi  7.595e+05  1.647e+04   46.114 < 2e-16 ***
## series6 Serisi  1.259e+06  4.761e+04   26.446 < 2e-16 ***
## series7 Serisi  1.920e+06  3.440e+04   55.830 < 2e-16 ***
## seriesi Serisi  7.262e+05  5.952e+04   12.201 < 2e-16 ***
## seriesM Serisi  2.128e+06  4.904e+04   43.403 < 2e-16 ***
## seriesZ Serisi  9.477e+05  5.043e+04   18.791 < 2e-16 ***
## segmentD Segment        NA          NA          NA          NA  
## segmentE Segment        NA          NA          NA          NA 
```

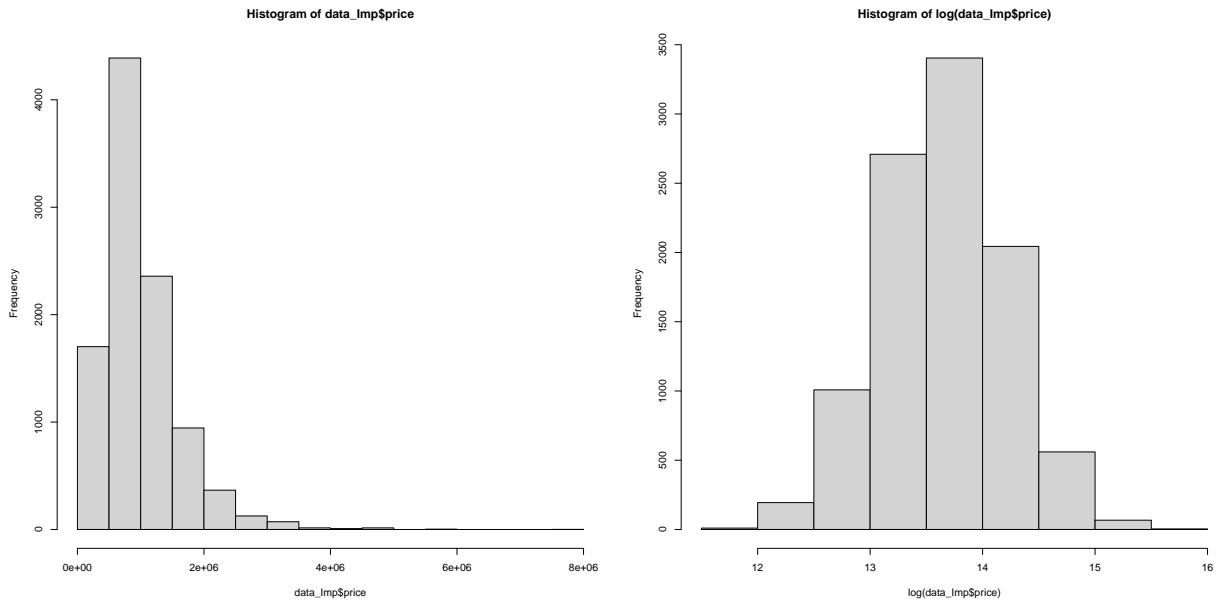
```

## segmentF Segment      NA      NA      NA      NA
## horsepower      7.326e+02  1.018e+02  7.193 6.89e-13 ***
## curb_weight     7.249e+00  3.446e+01  0.210  0.8334
## top_speed       2.055e+02  2.977e+02  0.690  0.4900
## model_year      8.692e+04  5.464e+02  159.089 < 2e-16 ***
## mileage        -7.773e-01  4.391e-02 -17.702 < 2e-16 ***
## is_heavy_damaged1 -4.198e+04  2.292e+04 -1.831  0.0671 .
## risk_score      -1.229e+03  1.871e+02 -6.567 5.44e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 222700 on 7983 degrees of freedom
## Multiple R-squared:  0.8588, Adjusted R-squared:  0.8585
## F-statistic:  3035 on 16 and 7983 DF,  p-value: < 2.2e-16

```



Bu grafiklerden görüldüğü üzere özellikle 1. ve 3. grafiklerdeki artık dağılımlarında iyi bir grafik dağılımı yok. Hafif u şekli var bunun için log dönüşümü yapacağız. Zaten verimiz sağa çarpık. Aşağıdaki grafik bunu göstermektedir. Ayrıca modelde singülerlik var bunu aşmak için NA olan p-değerlerine sahip olan değişkenleri çıkartarak değişken sayısını (boyutu) azaltacağız. Varyans homojenliği yoktur.



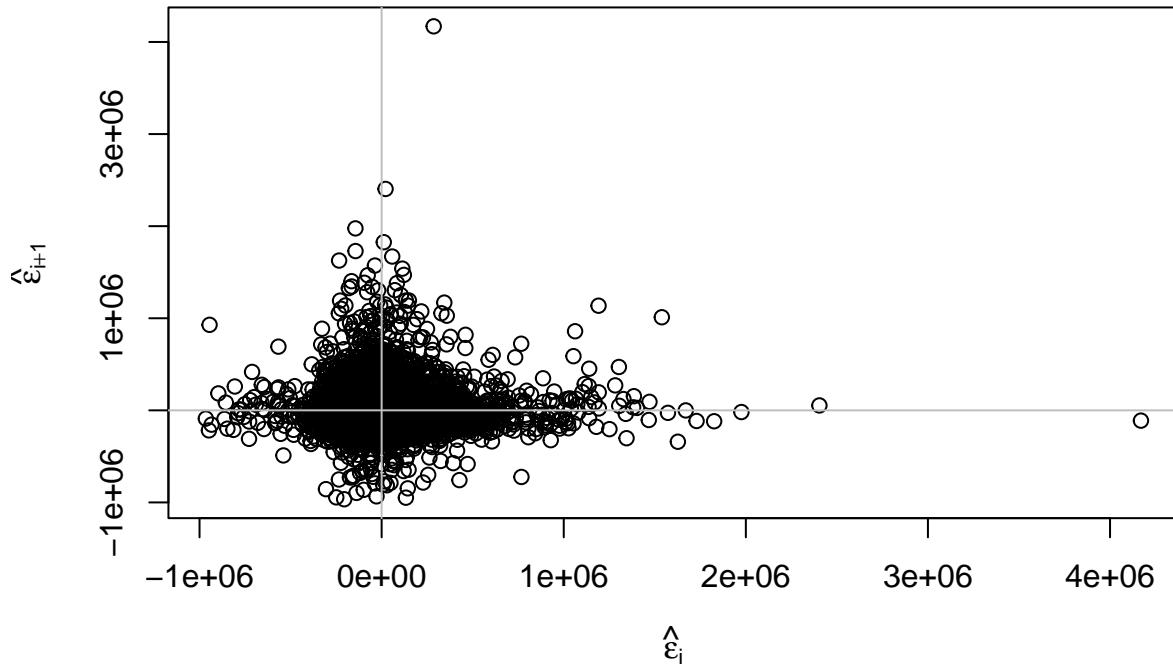
```
bptest(model_without_log)
```

```
##  
## studentized Breusch-Pagan test  
##  
## data: model_without_log  
## BP = 887.65, df = 16, p-value < 2.2e-16
```

```
dwtest(model_without_log)
```

```
##  
## Durbin-Watson test  
##  
## data: model_without_log  
## DW = 1.9692, p-value = 0.7603  
## alternative hypothesis: true autocorrelation is greater than 0
```

```
res = residuals(model_without_log)  
n <- length(res)  
plot(tail(res,n-1) ~ head(res,n-1),xlab = expression(hat(epsilon)[i]),  
     ylab = expression(hat(epsilon)[i + 1]))  
abline(h = 0, v= 0,col=grey(0.75))
```



```

cor(tail(res,n-1) , head(res,n-1))

## [1] 0.0153823

lillie.test(res)

##
##   Lilliefors (Kolmogorov-Smirnov) normality test
##
## data: res
## D = 0.10673, p-value < 2.2e-16

cvm.test(res)

## Warning in cvm.test(res): p-value is smaller than 7.37e-10, cannot be computed
## more accurately

##
##   Cramer-von Mises normality test
##
## data: res
## W = 35.803, p-value = 7.37e-10

```

```

test_res <- logical(1000)
for(i in 1:1000) {
  res_sample <- sample(res,50)
  test_res[i] <- shapiro.test(res_sample)$p > 0.05
}
table(test_res)

## test_res
## FALSE TRUE
## 791 209

```

Testlere bakıldığı üzere Durbin-Watson harici hiçbir varsayılmamaktadır. Şu an için tüm değişkenlerle modeli tekrardan bağımlı değişkeni `log()` dönüşümü ile kuracağız. Durbin-Watson değeri 1.5 - 2.5 arasında ve sağlanır. Yani grafikteki varyans homojenliğini olmamasını Breusch-Pagan test ile doğruluyoruz. Ayrıca artıklar da QQ-Plot ve testlerde görüldüğü üzere normal dağılmıyor. Grafikte görüldüğü üzere de artıklar arasında korelasyon yoktur.

`alias()` ile mükemmel bağımlılıkları görebiliyoruz. Segment ile seriler arasında var. Zaten seriler genelde belirli segment olduğundan bu beklenen bir durumdur.

```

alias(model_without_log)

## Model :
## price ~ series + segment + horsepower + curb_weight + top_speed +
##       model_year + mileage + is_heavy_damaged + risk_score
##
## Complete :
##             (Intercept) series2 Serisi series3 Serisi series4 Serisi
## segmentD Segment 0          0          1          1
## segmentE Segment 0          0          0          0
## segmentF Segment 0          0          0          0
##             series5 Serisi series6 Serisi series7 Serisi seriesi Serisi
## segmentD Segment 0          0          0          0
## segmentE Segment 1          0          0          1
## segmentF Segment 0          1          1          0
##             seriesM Serisi seriesZ Serisi horsepower curb_weight top_speed
## segmentD Segment 1          0          0          0          0
## segmentE Segment 0          1          0          0          0
## segmentF Segment 0          0          0          0          0
##             model_year mileage is_heavy_damaged1 risk_score
## segmentD Segment 0          0          0          0
## segmentE Segment 0          0          0          0
## segmentF Segment 0          0          0          0

```

2. Log dönüşümü ve Segment Çıkarılmış

Log dönüşümü uygulandı ve segment NA olan anlamsız değişken çıkarıldı ve model singülerlikten kurtarıldı. Artık VIF değerlerine de bakılabilir. Ama önce amacımız modeldeki anlamsız p değerine sahip olan değişkenleri çıkarmaktır ve performansına bakmaktadır.

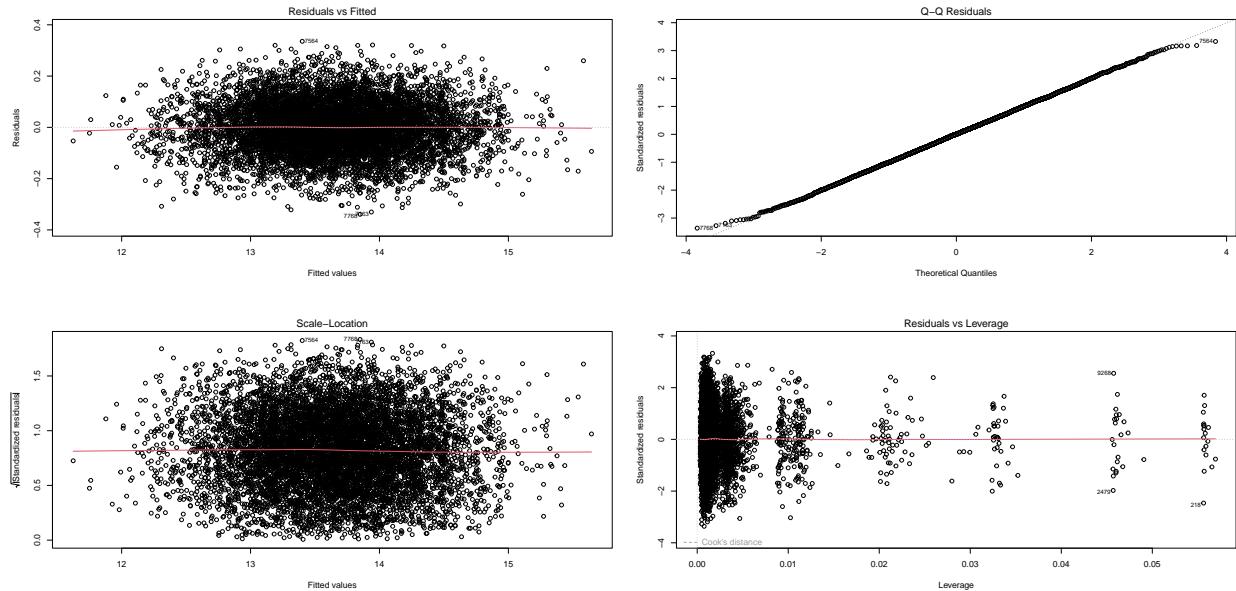
```

modelbase <- lm(log(price) ~ . -segment, data = trainset)
summary(modelbase)

## 
## Call:
## lm(formula = log(price) ~ . - segment, data = trainset)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -0.33914 -0.06862  0.00068  0.06749  0.33557 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -1.679e+02 5.006e-01 -335.352 <2e-16 ***
## series2 Serisi 1.040e-01 6.506e-03  15.984 <2e-16 ***
## series3 Serisi 5.052e-01 4.820e-03 104.814 <2e-16 ***
## series4 Serisi 7.090e-01 7.783e-03  91.089 <2e-16 ***
## series5 Serisi 9.071e-01 7.466e-03 121.487 <2e-16 ***
## series6 Serisi 1.206e+00 2.158e-02  55.879 <2e-16 ***
## series7 Serisi 1.516e+00 1.559e-02  97.205 <2e-16 ***
## seriesi Serisi 9.194e-01 2.698e-02  34.074 <2e-16 ***
## seriesM Serisi 1.508e+00 2.223e-02  67.829 <2e-16 ***
## seriesZ Serisi 9.991e-01 2.286e-02  43.701 <2e-16 ***
## horsepower     7.111e-04 4.617e-05 15.402 <2e-16 ***
## curb_weight    -1.670e-05 1.562e-05 -1.069  0.285  
## top_speed       2.016e-04 1.350e-04   1.494  0.135  
## model_year      8.977e-02 2.477e-04 362.439 <2e-16 ***
## mileage        -8.132e-07 1.991e-08 -40.853 <2e-16 *** 
## is_heavy_damaged1 -1.697e-03 1.039e-02  -0.163  0.870  
## risk_score      -1.623e-03 8.482e-05 -19.138 <2e-16 *** 
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 0.1009 on 7983 degrees of freedom
## Multiple R-squared:  0.9672, Adjusted R-squared:  0.9671 
## F-statistic: 1.471e+04 on 16 and 7983 DF,  p-value: < 2.2e-16

```

Varsayımların grafiklerimiz şuan gayet stabiller.



Modeli tüm anlamsızlardan arındırarak asıl modelimizi kuruyoruz.

```
model2 <- lm(log(price) ~ series + model_year + mileage +
               horsepower + risk_score,
               data = trainset)
```

```
summary(model2)
```

```
##
## Call:
## lm(formula = log(price) ~ series + model_year + mileage + horsepower +
##     risk_score, data = trainset)
##
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -0.34249 -0.06868  0.00114  0.06737  0.33692 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -1.678e+02  4.990e-01 -336.35 <2e-16 ***
## series2 Serisi  1.024e-01  6.254e-03   16.38 <2e-16 ***
## series3 Serisi  5.070e-01  3.776e-03  134.26 <2e-16 ***
## series4 Serisi  7.079e-01  6.178e-03  114.58 <2e-16 ***
## series5 Serisi  9.063e-01  4.404e-03  205.77 <2e-16 ***
## series6 Serisi  1.208e+00  1.988e-02   60.78 <2e-16 ***
## series7 Serisi  1.515e+00  1.204e-02  125.81 <2e-16 ***
## seriesi Serisi  9.256e-01  2.625e-02   35.26 <2e-16 ***
## seriesM Serisi  1.511e+00  2.090e-02   72.30 <2e-16 ***
## seriesZ Serisi  1.004e+00  2.214e-02   45.34 <2e-16 ***
## model_year      8.977e-02  2.476e-04  362.55 <2e-16 ***
## mileage        -8.130e-07  1.991e-08  -40.84 <2e-16 ***
## horsepower     7.128e-04  4.615e-05   15.45 <2e-16 ***
## risk_score     -1.627e-03  8.180e-05  -19.89 <2e-16 ***
## ---
```

```

## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1009 on 7986 degrees of freedom
## Multiple R-squared:  0.9672, Adjusted R-squared:  0.9671
## F-statistic: 1.81e+04 on 13 and 7986 DF,  p-value: < 2.2e-16

bptest(model2)

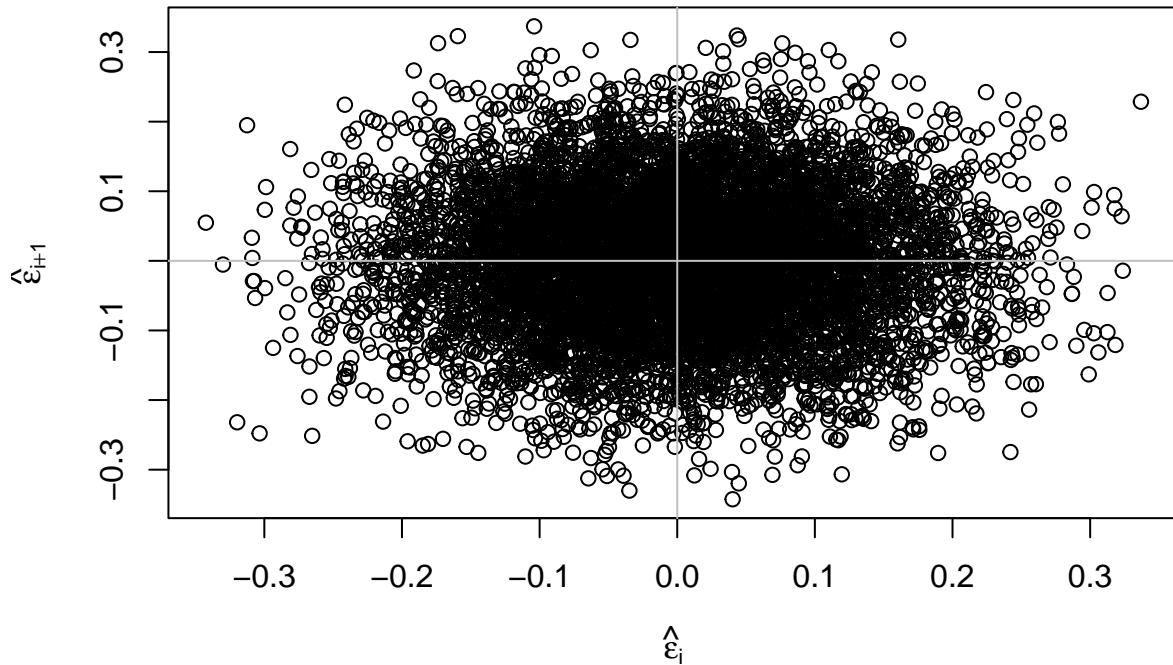
##
## studentized Breusch-Pagan test
##
## data: model2
## BP = 12.432, df = 13, p-value = 0.4925

dwtest(model2)

##
## Durbin-Watson test
##
## data: model2
## DW = 2.0044, p-value = 0.5775
## alternative hypothesis: true autocorrelation is greater than 0

res = residuals(model2)
n <- length(res)
plot(tail(res,n-1) ~ head(res,n-1),xlab = expression(hat(epsilon)[i]),
     ylab = expression(hat(epsilon)[i + 1]))
abline(h = 0, v= 0,col=grey(0.75))

```



```

cor(tail(res,n-1) , head(res,n-1))

## [1] -0.002208519

lillie.test(res)

##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data: res
## D = 0.0069758, p-value = 0.4531

cvm.test(res)

##
##  Cramer-von Mises normality test
##
## data: res
## W = 0.029675, p-value = 0.8518

test_res <- logical(1000)
for(i in 1:1000) {
  res_sample <- sample(res,50)

```

```

    test_res[i] <- shapiro.test(res_sample)$p > 0.05
}
table(test_res)

```

```

## test_res
## FALSE TRUE
##   39   961

```

Multicollinearity (collinearity) varsayımları için car paketinden `vif()` fonksiyonu ile Generalized Variance Inflation Factore baktık. Bunun sebebi değişkenlerde kategorik değişkenlerin olmasıdır. Serbestlik derecesine göre hesap yeniden yapılır ve görüldüğü üzere herhangi bir VIF problemi yoktur.

$$GVIF_j = \frac{\det(R_j) \cdot \det(R_{-j})}{\det(R)}$$

```
vif(model2)
```

```

##          GVIF Df GVIF^(1/(2*Df))
## series     3.008710  9      1.063106
## model_year 1.018366  1      1.009141
## mileage     1.014120  1      1.007035
## horsepower  2.995982  1      1.730891
## risk_score   1.032906  1      1.016320

```

```

predictionModel2 = predict(model2,testset)
predictionModel2 = exp(predictionModel2)

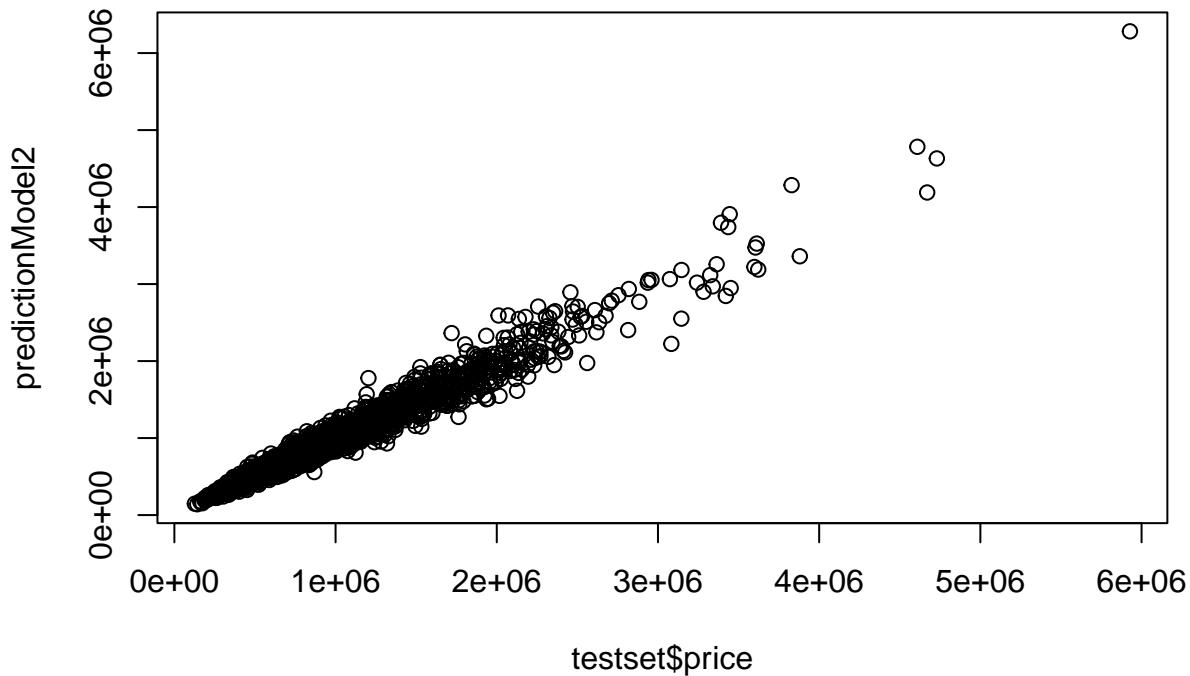
results_df <- data.frame(
  Model = c("predictionModel2"),
  R2    = c(
    R2(predictionModel2, testset$price)
  ),
  MAE   = c(
    MAE(predictionModel2, testset$price)
  ),
  RMSE  = c(
    RMSE(predictionModel2, testset$price)
  ),
  MAPE  = c(
    mean(abs((testset$price - predictionModel2) / testset$price)) * 100
  )
)
results_df

```

```

##           Model        R2       MAE       RMSE      MAPE
## 1 predictionModel2 0.9594803 83829.76 120005.6 8.508048

```



Modelin tüm testleri ve grafikleri güzel geldi. Yukarıda görüldüğü üzere gerçek tahmini değerler saçılım grafiği de gayet güzel. Ayıca varyanslar homojen residual normal dağılıyor. Plot yorumu: Component + residual plotlar, model year değişkeni için doğrusal varsayımin oldukça iyi sağlandığını; mileage, horsepower ve risk_score değişkenlerinde ise hafif doğrusal olmayan yapılar bulunduğuunu gösteriyor.

3. Modelden artık ve Cooks Distance yöntemleriyle artıkları ayırma

Buradaki amaç artıkları standartlaştırdı 2 alt ve üst standart sapmasını kaplayan alanın dışındaki değerleri almak ve cooks distance ile modeldeki aykırıları almak. Büyük olan ölçüt ve standartlaştırmış aykırıların 2 standart sapma dışında kalanların ortak indekslerini alıp veriden dışarı atıyoruz.

```
## model resiudallerinden outlier çıkarma
standardized_residuals = rstandard(model2)
summary(standardized_residuals)
```

```
##      Min.    1st Qu.     Median      Mean    3rd Qu.      Max.
## -3.3938822 -0.6812083  0.0113390  0.0000031  0.6684377  3.3396225
```

```
olcut1Index = which(abs(standardized_residuals)>2)
length(olcut1Index)
```

```
## [1] 367
```

```

dist = cooks.distance(model2)
olcut1 = mean(dist)*3

olcut2 = 4/length(dist)

olcut1;olcut2

## [1] 0.0003788724

## [1] 5e-04

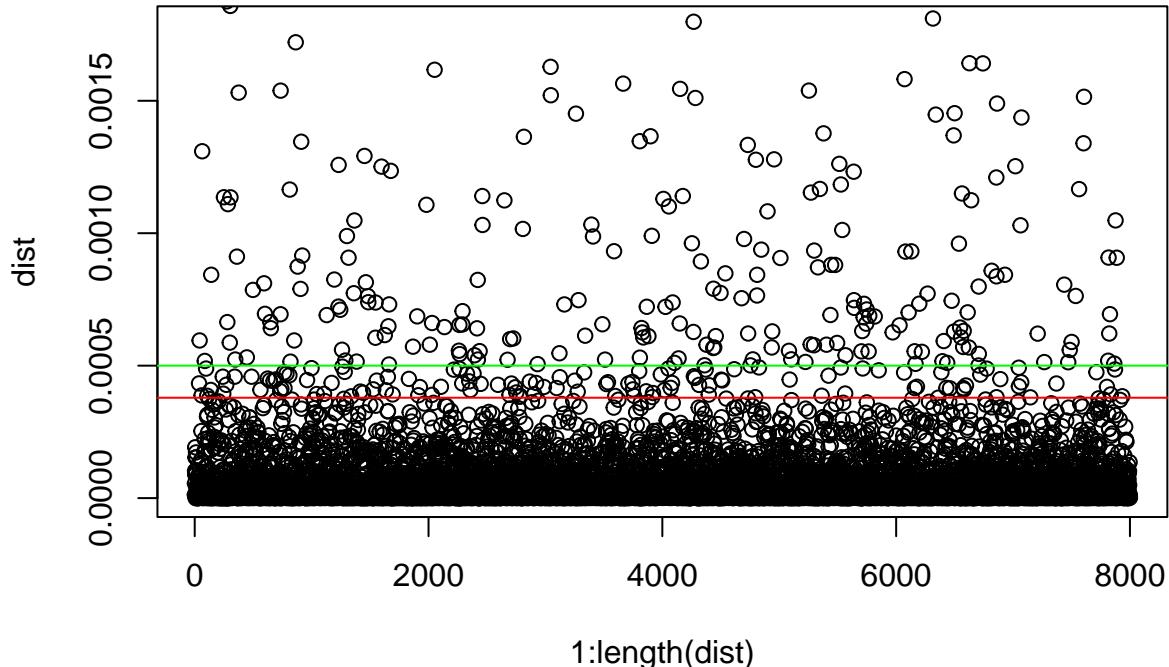
olcut1Index = which(dist>olcut1)
olcut2Index = which(dist>olcut2)
length(olcut1Index);length(olcut2Index)

## [1] 417

## [1] 294

plot(1:length(dist),dist,type = "p", ylim = range(dist)*c(1,0.07))
abline(h = olcut1, col = "red")
abline(h = olcut2, col = "green")

```



```

outliers = which(dist>olcut2 & abs(standardized_residuals)>2)

trainsetrem = trainset[-outliers,]
nrow(trainset);nrow(trainsetrem)

## [1] 8000

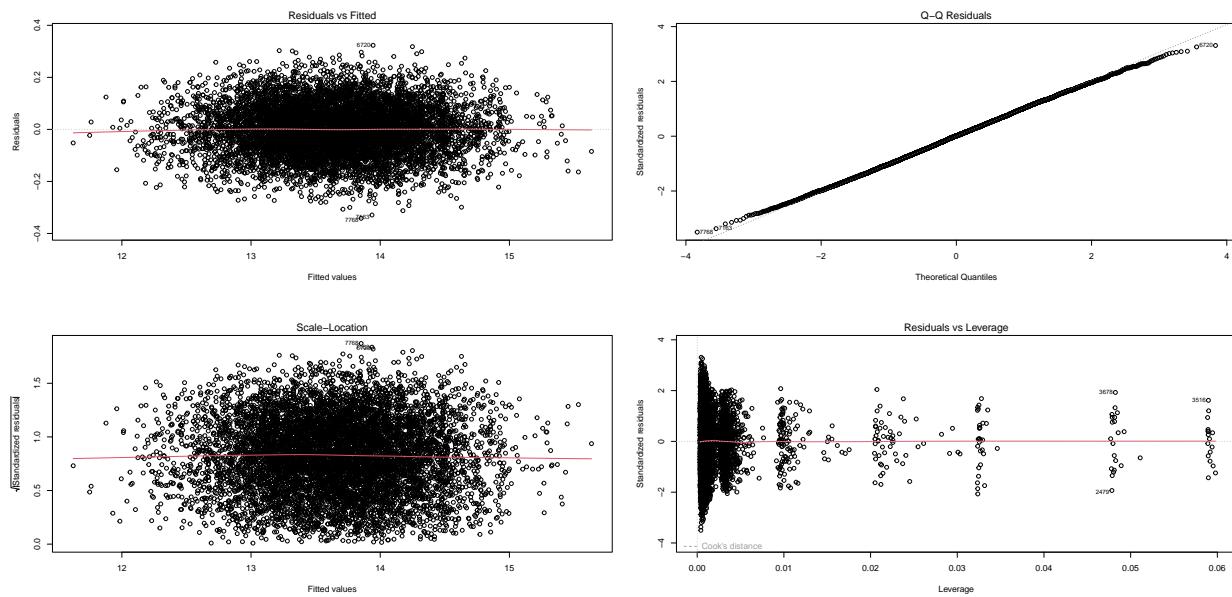
## [1] 7902

## model resiudallerinden outlier çıkarma

model3 <- lm(log(price) ~ series + model_year + mileage +
              horsepower + risk_score,
              data = trainsetrem)
summary(model3)

## 
## Call:
## lm(formula = log(price) ~ series + model_year + mileage + horsepower +
##     risk_score, data = trainsetrem)
##
## Residuals:
##       Min     1Q   Median     3Q    Max 
## -0.34170 -0.06725  0.00138  0.06687  0.32310 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -1.676e+02  4.862e-01 -344.73  <2e-16 ***
## series2 Serisi  1.043e-01  6.189e-03   16.86  <2e-16 ***
## series3 Serisi  5.064e-01  3.678e-03   137.69 <2e-16 ***
## series4 Serisi  7.072e-01  6.094e-03   116.05 <2e-16 ***
## series5 Serisi  9.060e-01  4.295e-03   210.96 <2e-16 ***
## series6 Serisi  1.207e+00  1.926e-02    62.64 <2e-16 ***
## series7 Serisi  1.510e+00  1.195e-02   126.35 <2e-16 ***
## seriesi Serisi  9.369e-01  2.604e-02    35.97 <2e-16 ***
## seriesM Serisi  1.502e+00  2.052e-02    73.21 <2e-16 ***
## seriesZ Serisi  9.908e-01  2.189e-02    45.26 <2e-16 ***
## model_year      8.965e-02  2.413e-04   371.60 <2e-16 ***
## mileage        -8.173e-07  1.951e-08   -41.89 <2e-16 ***
## horsepower      7.261e-04  4.515e-05    16.08 <2e-16 ***
## risk_score      -1.653e-03  8.163e-05   -20.25 <2e-16 ***
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09762 on 7888 degrees of freedom
## Multiple R-squared:  0.969, Adjusted R-squared:  0.9689
## F-statistic: 1.895e+04 on 13 and 7888 DF, p-value: < 2.2e-16

```



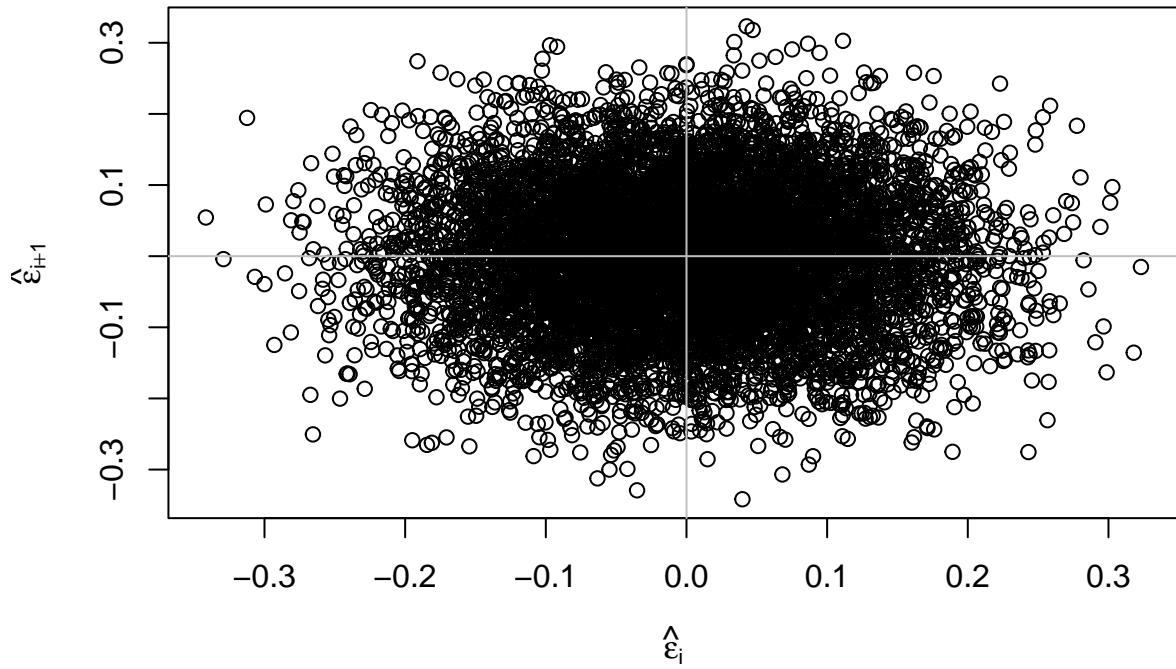
```
bptest(model3)
```

```
##
## studentized Breusch-Pagan test
##
## data: model3
## BP = 49.121, df = 13, p-value = 4.216e-06
```

```
dwtest(model3)
```

```
##
## Durbin-Watson test
##
## data: model3
## DW = 2.012, p-value = 0.7038
## alternative hypothesis: true autocorrelation is greater than 0
```

```
res = residuals(model3)
n <- length(res)
plot(tail(res,n-1) ~ head(res,n-1), xlab = expression(hat(epsilon)[i]),
      ylab = expression(hat(epsilon)[i + 1]))
abline(h = 0, v= 0,col=grey(0.75))
```



```

cor(tail(res,n-1) , head(res,n-1))

## [1] -0.00604867

predictionModel3 = predict(model3,testset)
predictionModel3 = exp(predictionModel3)

results_df <- data.frame(
  Model = c("predictionModel3"),
  R2   = c(
    R2(predictionModel3, testset$price)
  ),
  MAE  = c(
    MAE(predictionModel3, testset$price)
  ),
  RMSE = c(
    RMSE(predictionModel3, testset$price)
  ),
  MAPE = c(
    mean(abs((testset$price - predictionModel3) / testset$price)) * 100
  )
)
results_df

##          Model        R2       MAE      RMSE     MAPE
## 1 predictionModel3 0.9594337 83763.36 120036.8 8.507456

```

Karşılatırmalı Grafik

```
results_df <- data.frame(
  Model = c("predictionModel3", "predictionModel2"),
  R2     = c(
    R2(predictionModel3, testset$price),
    R2(predictionModel2, testset$price)
  ),
  MAE   = c(
    MAE(predictionModel3, testset$price),
    MAE(predictionModel2, testset$price)
  ),
  RMSE  = c(
    RMSE(predictionModel3, testset$price),
    RMSE(predictionModel2, testset$price)
  ),
  MAPE  = c(
    mean(abs((testset$price - predictionModel3) / testset$price)) * 100,
    mean(abs((testset$price - predictionModel2) / testset$price)) * 100
  )
)
results_df

##           Model      R2      MAE      RMSE      MAPE
## 1 predictionModel3 0.9594337 83763.36 120036.8 8.507456
## 2 predictionModel2 0.9594803 83829.76 120005.6 8.508048
```

4. Final Model Seçimi

Grafiklerde model varsayımlarında problem görünmüyör fakat Breusch-Pagan test sağlanmamaktadır. Başka testlere bakılabilir. Ayrıca model2 MAPE ve MAE dışında diğer iki skorda öndedir. model2'nin R2 ve RMSE'si daha iyidir. Varsayımları sağlamlığından dolayı model2 yi tercih ediyoruz.

Modeli scale edip baktı fakat skorlarda fark yoktu. Final modelimiz model2'dir.

Gerçek veri setinden tahmin örneği

Burada gerçek modelin tahmin performansı grafikleri ve varsayımları testlerini görüyoruz modelimiz görmediği arabayı yaklaşık 100 bin TL farkla tahmin etti. Fakat bu lineer model bazı varsayımları sağlamıyor grafikler iyi olsa bile bu yüzden genel sunum verisi bu veriden üretilmiş sentetik veriydi. Bu bölüm ekstra gerçek fiyatlarla kurulmuş modelin tahmin performansını göstermek için yazıldı.

```
data_clean <- read.csv("data_clean_seri.csv")
data_clean$series <- as.factor(data_clean$series)
set.seed(301)
imputed = mice(data_clean, method = "rf", printFlag = FALSE, m = 3)
data_Imp = complete(imputed, 3)
set.seed(301)
sampleIndex <- sample(1:nrow(data_Imp), size = 0.8 * nrow(data_Imp))
trainset <- data_Imp[sampleIndex,]
testset <- data_Imp[-sampleIndex,]
```

```

model <- lm(log(price) ~ series + year +
             kilometer +
             hp + damaged + risk_score,
             data = trainset)
standardized_residuals = rstandard(model)
olcut1Index = which(abs(standardized_residuals)>2)
dist = cooks.distance(model)
olcut1 = mean(dist)*3
olcut2 = 4/length(dist)
olcut1Index = which(dist>olcut1)
olcut2Index = which(dist>olcut2)
outliers = which(dist>olcut2 & abs(standardized_residuals)>2)
trainsetrem = trainset[-outliers,]
train_means <- apply(trainsetrem[,c(-1,-5,-7)], 2, mean)
train_sds <- apply(trainsetrem[,c(-1,-5,-7)], 2, sd)
train_scaled <- trainsetrem
test_scaled <- testset
train_scaled[,c(-1,-5,-7)] <- scale(trainsetrem[,c(-1,-5,-7)], center = train_means, scale = train_sds)
test_scaled[,c(-1,-5,-7)] <- scale(testset[,c(-1,-5,-7)], center = train_means, scale = train_sds)
modelx <- lm(log(price) ~ series + year +
              kilometer +
              hp + damaged + risk_score,
              data = train_scaled)

```

```
summary(modelx)
```

```

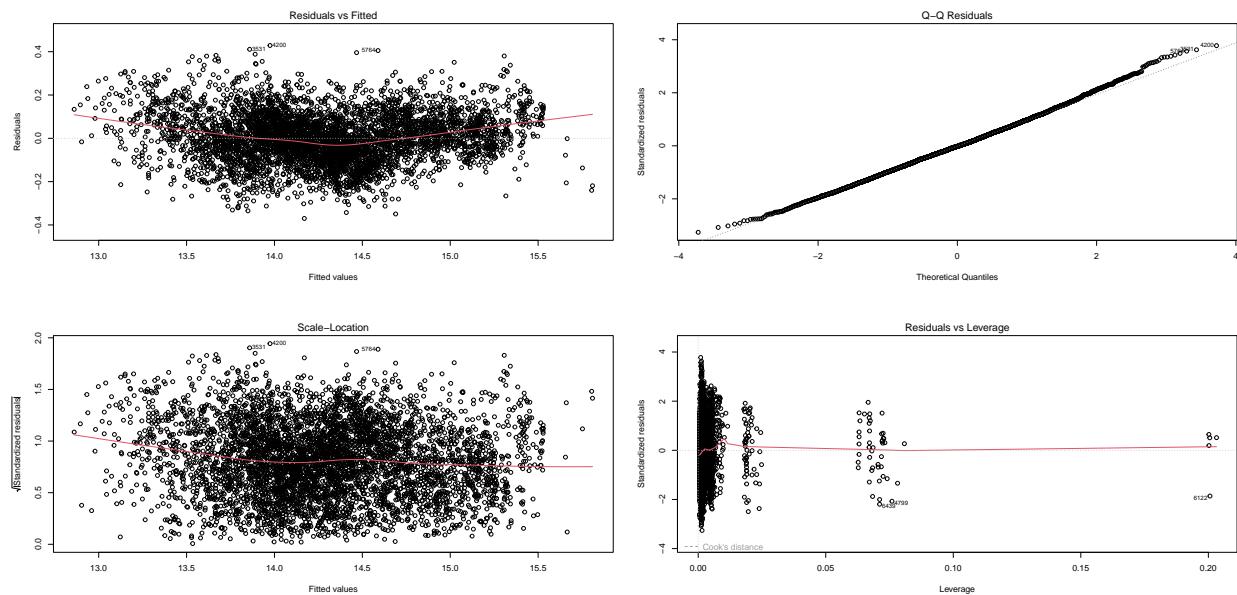
##
## Call:
## lm(formula = log(price) ~ series + year + kilometer + hp + damaged +
##      risk_score, data = train_scaled)
##
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -0.37002 -0.07580 -0.00245  0.07325  0.42868 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 14.043156  0.005003 2806.943 < 2e-16 ***
## series2 Serisi  0.029645  0.009375  3.162  0.00158 ** 
## series3 Serisi  0.199197  0.005422 36.739 < 2e-16 ***
## series4 Serisi  0.364172  0.008976 40.570 < 2e-16 ***
## series5 Serisi  0.423169  0.006458 65.524 < 2e-16 ***
## series6 Serisi  0.701763  0.032949 21.299 < 2e-16 ***
## series7 Serisi  0.796366  0.018693 42.603 < 2e-16 ***
## seriesi Serisi  0.734597  0.053830 13.647 < 2e-16 *** 
## seriesM Serisi  0.285036  0.037771  7.546 5.28e-14 ***
## seriesZ Serisi  0.518881  0.030410 17.063 < 2e-16 *** 
## year          0.321566  0.002638 121.907 < 2e-16 *** 
## kilometer     -0.146821  0.002640 -55.615 < 2e-16 *** 
## hp            0.094706  0.002666 35.526 < 2e-16 *** 
## damaged        -0.127061  0.007497 -16.947 < 2e-16 *** 
## risk_score    -0.033822  0.001735 -19.489 < 2e-16 *** 
## ---

```

```

## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1135 on 5037 degrees of freedom
## Multiple R-squared:  0.9521, Adjusted R-squared:  0.952
## F-statistic:  7159 on 14 and 5037 DF,  p-value: < 2.2e-16

```



```
bptest(modelx)
```

```

##
## studentized Breusch-Pagan test
##
## data: modelx
## BP = 197.27, df = 14, p-value < 2.2e-16

```

```
dwttest(modelx)
```

```

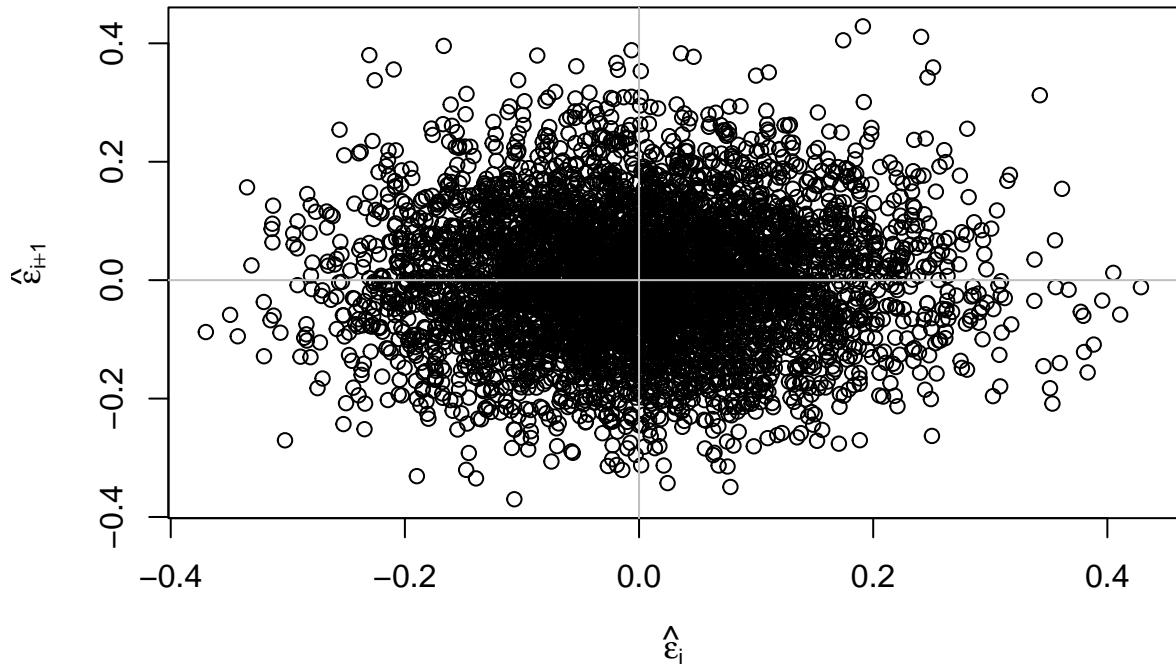
##
## Durbin-Watson test
##
## data: modelx
## DW = 1.9705, p-value = 0.1471
## alternative hypothesis: true autocorrelation is greater than 0

```

```

res = residuals(modelx)
n <- length(res)
plot(tail(res,n-1) ~ head(res,n-1), xlab = expression(hat(epsilon)[i]),
      ylab = expression(hat(epsilon)[i + 1]))
abline(h = 0, v= 0,col=grey(0.75))

```



```
cor(tail(res,n-1) , head(res,n-1))
```

```
## [1] 0.0142182
```

Gerçek internetten alınmış ilan verisinden tahmin.

```
carBMW <- data.frame(series = "3 Serisi", year = 2016, kilometer = 197000, hp = 190, risk_score = 86,dan
carBMW[,c(-1,-6,-7)] <- scale(carBMW[,c(-1,-6,-7)], center = train_means, scale = train_sds)
predOne = predict(modelx,carBMW)
predOne = exp(predOne)
cat("Gerçek Fiyat:",1495000,"\\nTahmin Edilmiş Fiyat:",predOne)
```

```
## Gerçek Fiyat: 1495000
## Tahmin Edilmiş Fiyat: 1411480
```

```
# https://www.sahibinden.com/ilan/vasita-otomobil-bmw-gunaydin-garajdan-1285567182/detay
```

Gerçek veriden eğitilmiş model için test skorları aşağıdaki gibidir.

```
predictionModelX = predict(modelx,test_scaled)
predictionModelX = exp(predictionModelX)

results_df <- data.frame(
```

```

Model = c("predictionModelX"),
R2    = c(
  R2(predictionModelX, test_scaled$price)
),
MAE   = c(
  MAE(predictionModelX, test_scaled$price)
),
RMSE  = c(
  RMSE(predictionModelX, test_scaled$price)
),
MAPE  = c(
  mean(abs((testset$price - predictionModelX) / testset$price)) * 100
)
)
results_df

```

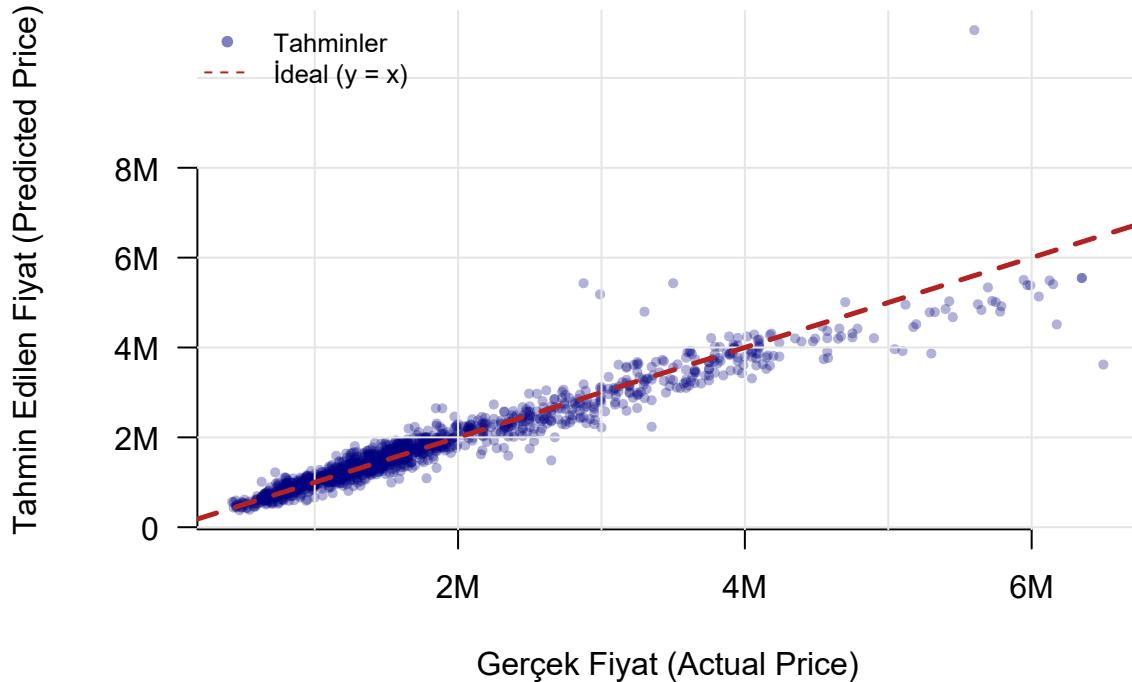
```

##           Model      R2      MAE      RMSE      MAPE
## 1 predictionModelX 0.9184741 179326.6 315597.2 9.875926

```

Gerçek fiyat ve tahmin edilmiş fiyat grafiği. Bazı gözlemlerimiz görüldüğü üzere aykırı değer ve RMSE'yi aşırı yükseltiyor.

Model Tahmin Başarısı



Her ne kadar belirlediğimiz influencial gözlemleri çıkarmış da olsak orada elediğimiz verilerden sonra modelimizin leverageleri daha kötü oldu. Bu yüzden o modelimizi buraya koymadık ve herhangi bir ek varsayımi sağlamadık. Burada yapılabilecek şey farklı en küçük kareler (OLS) regresyon dışındaki daha farklı yöntemlere geçmektir. Ridge, Lasso, Elastic Net gibi cezalandırma yöntemlerinin bulunduğu ve açıklanabilirliğin

hala olduğu regresyon yöntemleri veya daha esnek yöntemlere geçmek modelimizin performansı ve geçerli olması için önemli bir adım olacaktır. Burada residual dağılımındaki özellikle çizgideki U-şekilli yapı lineer olmayan değişkenlere işaret olabilir. Bu durum doğrusal regresyon varsayımlarını ihlal eder ve bu noktada polinom terimler ya da daha az açıklanabilir olan Random Forest, Gradient Boosting ya da Gradient Boosting Decision Trees (LightGBM, XGBoost, CatBoost gibi) modellere geçilmesidir. Bu veriyi şuan en iyi tahmin eden model bu yazı dışında geliştirilmiş ve eğitilmiş olan CatBoost modelidir.