ML Recap\_27\_07\_2023

Auto Scout dataseti--> otomobil fiyatlarini tahmin etmeeya calisacagiz

#model optimization with Grid Search

#hangi modeli sececegimiz datasetinin tipine bagli, biz genelde birden fazla algortima ile calisip kiyaslayarak modelimizi seciyoruz.

#daha fazla data olsa model iyilesir seklinde onerilerde de bulunabilirsiniz

#en son asamada deployment asamasidir.

**Tasks**

1. Import modules
2. EDA
3. Train/Test split
4. Implement Linear Regression
5. Implement Ridge Regression
6. Implement Lasso Regression
7. Implement Elastic Net
8. Visually Compare Models Performance In a Graph
9. Feature Importance
10. Compare Models Performance
11. Prediction

#Birden fazla notebook olacak, deneyecegiz. Standard scaleri mesela, min mx scaler’i.

Default halde birakirsaniz default 10 mesela 10 tane satir gelir.

Price-->target

Cons\_comb--> sehir ici ve disi yakit kullaniminin birlestiren numeric bir sutun

Std, mean’e yakinsa orda bir outlier sorunu vardir.

Min ve max arsinda, ya da %25’lik %75’lik kisimlarinda cok ucurum varsa, yine outlier var diyebiliriz.

Bazen de binary sutunlarda bu kiyaslamayi yapmam zor olur, cunku std mena’e yakin olabilir 1-0 seklindeki bir datada.

Describe icine include=all ya da include=object yazarsan numeric haricidnekileri de getirebilir.

**Feature Engineering**

Air conditioning, cruise control, climate gibi cesitli fonksiyonlar var. Bunlari ayirabiliriz mesela. Bunlari ayiracagiz ileride.

**Dummy control**

Audi A2 bir tane oldugu icin drop ediyoruz.

23 feature var su an

**Hisplota** bakinca hafifi bir saga carpiklikla normal dagilim diyebiliriz.. price’a baktik tabi bunun icin.

**Numeric degerler:**

Toplam 10 tane sutunumuz target dahil numeric mis

**Heatmap**

Linear regresyon uygulayabilmemiz icin, target ile anlamli bir iliski olmasi gerekiyor.

numeric feature’lar ile target arasinda korelasyona bakiyoruz.

**Multicolinearity**

Feature’lar arasinda yuksek korelasyon var mi diye bakiyoruz. Bir kod satiri var bunun icin. Bu sorun varsa, Ridge ve Lasso gibi regularizasyon algoritmalarini kullanarak cozmeye calisiyoruz. Literatur ML icin 0.9 ustu multicolinearity kabul ediliyor.

**Boxplot**

Boxplot ile outlier’a bakiyoruz. Datanin genelinde degil de boyle alt gruplarla outlier’a bakiyoruz.

Whis default=1.5 idi, ama burda whisker=3 yaptik.

Yine de Audi A3’te cok outlier var.

Risge lasso ve elastic net hem outlier sorunun cozmek icin hem de multicolinearity sorunun cozmek icin kullaniliyor zaten.

10 tane birbiri ile iliskili feature’um var diyelim, belki icinden birkac tanesini sececegiz.

Feature selection yaparak bunu cozecegiz. Zaten ridge ve lassoyu feature selextion oiicn kullaniyoruz temelde.

Whis=1.5 gore outlier olabilecek degerleri gormemizi saglayan bir kod yazdirdik.

**Get dummies**

**Unique degerleri birbirinden ayirmak icin bir kodumuz var.**

Air conditioning, cruise control, climate gibi cesitli fonksiyonlar var. Bunlari ayirabiliriz mesela. Bunlari ayiriyoruz bu kodla. Sonra bunlara get\_dummies yapiyoruz. Ama bizim dummy tuzagi diye bir sorunumuz vardi. Bunun icin drop first yapiyoruz ornegin. Boylece de model complexity’i azaltmis oluyoruz. Yoksa fazla feature ile model iyilesmis gibi de gorunebilir ama overfittinge neden olan yalanci bir iyilesmedir bu aslinda. Ondan adi dummy tuzagi.

3 feature var diyelim, aracimiz LPG diyelim LPg’ye 1 yaziyor digerlerine 0 yaziyor

Diesel olsa diesel’e 1 yazacak digerlerine 0 yazacak. Digeri otomatik olarak kodlanmis olacak, yani benzinli. 3 tane ise 2 tanesini kodlamam yeterli

Elimizde 2 tane unique deger varsa, benzine 1, gaza 0 verecek. Bir kere kodlamam yeterli.

Get\_dummies yapinca artik object sutunumuz yok, hepsi artik numeric. Modele hazir.

#Ordinallik varsa Get\_dummies yapinca modeli yaniltir. Mesela aralarinda bir siralama var, pahali, ucuz, orta gibi

ordinal encoder kullanacagiz ordinal bir data varsa.

Arada ordinallik varsa, ona gore bir encoding kullanacagiz.

One\_hot encoder mesela yine get\_dummies yontemi-->hepsine ayni onemi veriyor.

**Corr**

Price ile korelasyonlarina bakiyoruz.

Vites ve horse power en korele gorunuyor.

Gorsellestirdigimizde de negatif ve pozitif korelasyon gorunuyor.

**Train-test split**

Bir kismini egitecegiz, diger kismini da testte kullanacagiz datamizin.

Egitim kismina gecmeden targeti kaldiriyoruz, drop ediyoruz.

Train-test ayrimi yaparken datasetine gore farklilik gosterebiliyor yaklasimimiz.

Bazilarinda randomlik istemeyebiliriz. Shuffle diye bir secenek var mesela- datayi karistirip randomize bir ayrim yapiyoruz mesela

Bazen de randomoze degil de ilk 100 satiri train olarak al, kalani test olarak al da diyebilriz.

Modeli kurarken zaten random\_state kullaniyoruz. Final modelde kullanmayacagiz zaten.

ama cesitli random\_stateler de kullanilarak kiyaslanabilir tabi.

Modelleri kiyaslarken de onemli random state yapmak, bir sabitlik sagliyoruz ki optimal duzeyde bir kiyaslama yapabilelim hangi modelim daha basarili diye.

Datanin buyuklugune gore degisebilir, test\_size=0.2 diyebiliriz.

Test\_size i degistirerek daha iyi sonuclar alabilirsiniz ama temel yaklasimlar 02, 0.3, 0.25

**Linear Regresyon**

Simple

Multiple--> birden fazla feature varsa

Features-label arasinda korelasyon olmali

Verilerin dagilimi normal olmali linear icin

Datasetinin bir kisminda iyi sonuclar aldik diyelim ama baska yerinden kotu sonuclar aldiysam, genelleme yapamiyoruz demektir.

Idealimiz dusuk bias ve dusuk variance’a sahip olmasi modelin

Bunun da bir optimumu var. Optimum model complexity.

Yuksek variance varsa-->overfitting

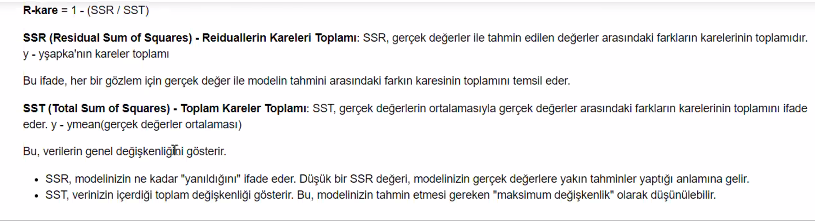
Yuksek bias--> underfitting

Modeli linear regresyona gore fit ediyoruz.

Train\_val--> icine fit edilmis modeli veriyoruz. Train ve test datasinin veriyoruz.

Benim featurelarim %89 oraninda aciklayabiliyor demektir.

verilerimizModelimize uymuyorsa R2 negatif gelebilir. Asagidaki formul dolayisiiyla



**Adjusted R2**

Bu da 0.88 cikti. Cok fazla fark yok digeri ile. Muhakkak cok iyilestirme yapacak diye bir sey yok.

**CV**

Datanin farkli yerlerinden alip test ediyoruz.

Bunlarin ortalamasina gore overfitting var demek ancak mumkun.

Grid searchun icinde zaten CV var. Extra yapmamiza gerek yok.

Return\_train\_score=True dersen train scorelarini da sana dondurur.

Neg\_mean...

Basina neden neg yaziyoruz.

Mean absolute errorlari minimize etmeye calisiyoruz. Neg yaziyorum ki max yapinca negatif bir seyi minimize etmis oluruz.

CV olmadan aldigimiz train\_val ile overfitting gozukmuyor demistik.

Ama her zaman CV sonrasi skorlara bakmamiz gerekecek.

RMSE’nin buyuklugu aslinda onemli degil. Cunku label’in degeri arttikca bu degerler de artar. Bu ne icin onemli? Overfitting kontrolu icin onemli bu.

Ben ortalamasini alinca modelimiz ne kadar hata yapiyor onu veriyor.

**Prediction Error**

Simdi graphize edecegiz. Yellow brick kutuphanesi bunlardan biri. Residuallarin dagilimini gosteriyor. Hem test hem train icin.

Residual=gercek deger-tahmin edilen deger.

2 cizgi var. 1. identity line 2. best fit line

Y’de y-pred var, x’de gercek degerler var.

Silik olan cizgi modelin mukemmel olan cizgi. R2=1, alttaki ve ustteki residual toplami 0

Best-fit line ne kadar yakinsa, modelimiz o kadar basarili diyebiliriz.

Aradaki fark acildikca modelimizin basari orani dusuyor diyoruz.

Residuallar bir pattern sergilememeli-->normal dagilmali

Residual plot ile buna bakiyorum. Pattern sergilerse linear regresyon icin uygun degil diyoruz.

Yellow brick sag tarafta residuallarin normal dagilima sahip olup olmadingi gosteriyor.

**Dropping Observations**

35binden buyuk olanlara bakalim bir dedik.

Bir de marka bazinda baktik bunlara

35binden buyuk olanlar haricindekileri df3’e atadaik.

Dusurduk bu datalari

Modeli tekrar egitiyoruz,

Baktim best-fit line ile identity line daha yakin

Modelimizin basarisi bir miktar artmis

R2 ile bakinca, RMSE ile bakinca da bir miktar iyilesme var.

Ozellikle saglik verilerinde onemli azicik duzelmeler bile

Yukarida %13’luk bir hata aliyorduk, simdi modeldeki hata orani ise %11

400 kusur veri dusurunce yani oldu bunlar

Pozitif kaysayi--> degisken arttikca target da artiyor.

**Pipeline**

Hata yapma riskimizi azaltiyor

Veri sizintisi riskini azaltiyor.

Daha temiz bir kod

Hem model olusturabiliyoruz, scale verebiliyoruz, CV ve Grid search yapabiliyoruz.

Operations icine scaler ve modelimizi veriyoruz.

**Sonra pipeline’n icine steps=operations’i vererek pipeline’i kuruyoruz.**

Icerisine encoding de olabilir bunun

Bu olusturdugumuz pipeline modelini fit ediyoruz.

Skorlarimizi aliyoruz.

Zaten burda aldigimiz skorlar da R2

Cross validation in pipeline

Fit isleminin yerine CV islemini yapiyoruz.

Icine pipe\_modeli veriyoruz, train datasini veriyoruz, scoring’i veriyoruz.

Score’lari ortalamasini alarak getiriyor bu.

**Implementing Ridge Regression**

**Ridge-lasso ve elastic net in Temel amaclari:**

1.overfittingi onlemek

2.Feature selection

3.Multicolinearity sorununu cozmek

Feature selection yapacaksak daha cok lasso

Overfitting ile mucadele ediyorsak daha cok ridge denebilir

Elastic net ise-->icindeki l1 l2 ratio’ya gore ridge veya lassoyu seciyor

**Scaling**

Ridge ve lasso yapilacaksa--> scaling yapmamiz gerekiyor.

Minmax’i tercih ettik get\_dummies yaptigimiz icin

Duruma gore scaleri seceriz.

**Once scaleri tanimladik**

**Scaler ile traini fit ettik**

**Sonra hem train hem test i scaler ile transform ettik**

**Model**

Ridge\_modelin icinde alpha var. Duzenleme kuvvetinin ifade ediyor.

1’e yaklastirarark overfittingi asmaya calisiyoruz

Grid search ile optimal alphayi tespit edecegiz zaten

Train\_val ile skorlari aliyoruz.

**Finding best alpha for Ridge**

Grid search ile en iyi hyperparametreleri buluyoruz.

Ama abartmamak lazim, cok maliyetli grid search

Burda oynayacagimiz parametre alpha parametresi, en iyi skoru veren alpha degerini bulacak

Param\_grid ile--> alpha\_space’i verdik buna. Olusturduk bunu

Estimator olarak ridge modeli verdik.

Scoring’e tek bir skor yazdik, neden digerlerini yazmadik? -->

CV’de istedigimiz kadar metric yazabiliyorduk ama

Burda benim verecegim arametre icin en optimal degeri bul diyoruz. Search et ve bana getir demis oluyoruz. O yuzden tek bir skor verebiliyoruz grid search’e

Default olarak R2 scoreu alir, yazarsak da tek metric yazacagiz.

CV yapar ve ortalamasi en yuksek olani dondurur.

Best\_params --> en iyi parametreyi dondurur.

Best\_estimator-->sadece default degerden farkli bir deger varsa onu dondurur.

Best\_score--> grid search icine yazdigimiz RMSE’nin best score’u

**LASSO**

Feature selection icin ozellikle kullanilir. Feature sayisini dusurmus oluyoruz.

Lasso modelin icine de scaled edilmis traini veriyoruz.

**Finding best alpha for Lasso**

Alpha\_space’i burda da kullaniyoruz. Ridge adimlarinin aynisini kullaniyoruz.

Grid search icinde coef yok, o yzuden aldigmiz parametreler ile modeli yeniden kuruyoruz

Katsayi km icin neagatif mesela--> ters iliski aracin fiyatiyla

Bircok feature’u sifirlamis, epeyini de sifira yaklastirmis

Hangi feature’larin katsayisi yuksekse ona gore feature selection yapiyoruz

**Elastic ne**t

l1\_ratio 0 secersek Ridge gibi davraniyor. Genelde lasso egilimlidir.

Hem feature selection hem multicolineraty sorunu var ise o zaman elastic\_net seceriz.

Yine aynisini uyguluyoruz elastic net icin

Skorlara bakinca, dusmus gorunuyor. Ama grid search ile de bakmak gerek

**Finding best alpha for elastic net**

Alha ve l1 ratio icin cesitli alternatifler sunuyoruz modele.

Iclerinde en iyisini bul getir demis oluyoruz.

Ilk basta l1 ratio olarak 0.5 ve ustunu vererek denemis olduk, sonuca gore belki 0.5’in altinda da verebiliriz.

Genelde lasso egilimli oldugu icin 0.5 ustunu ilk denemek mantikli

0.5 cikarsa--> ortasi gibi davraniyor.

L1 ratio 1 gelmis--> demekki lasso gibi davraniyor.

Grid search yapinca zaten modelin kotu olmadigini gorduk elastic net ile kurdugumuz

Basta skorlarim dustu gibi gorunuyordu.

CV yaparken de 10 yapmak yeterli.

**Feature Importance**

Yellow brick kutuphanesi bize katsayilara gore oneminin getiriyor

Bunlardan karar veriyorum ve 6 tane feature almaya karar verdim

Km, make\_model, horse power, age, gearing type, price

Onemli olan model complexity azaltmak

**Tahminlemeye etkisi en fazla olan featurelari seciyorum, bunlarin absolute buyukluklerine gore etkisine baki karar verdim**

Simdi kiyaslama yapacagim

35 binden yuksek olanlari devre disi birakarak yukaridaki ismlemleri burda da yaiyorum.

Lasso\_final model olarak train\_val’ine bakiyorum

Hata %11’den %12’ye cikmis

Bir miktar metricler dusmus

Linear

Ridge

Lasso

Elestic

Final model olmak uzere 5 tane modelim var. Bunlari kiyaslayacagimm

Tum skorlari toplu gorebilecegim bir kod var, onunla skorlari getirtitriyorum,

Final model R2 dusuk gorunuyor. 30 feature ile 0.90 olmasi yerine

6 sutunla %87 elde etmek onemli. Modelim basit, %87 fenal degil.

6 sutunla %12 hata mi, 30 feature ile %11 hata mi?

Tabi ki 6 sutunu sececegiz.

**Prediction**

Bir deger aldik diyelim, bunlari da get dummy yammaiz gerek

Ayni sekilde 0-1’e cevirerek bir tahmin aldik

**Prediction with random sample**

Randomize bir data alalim, bakalim nasil tahmin etmis dedik

Random\_samples’dan 20 tane sample aldim

Price’i dusurduk

Final scaler’i kullandik

Bunlardan prediction aldik

True\_labels olarak da gercek degerlerini aldik bunlarin

Modelimizin bunlari nasil tahmin ettgini de getiri bu ikisini kiyaasladik.