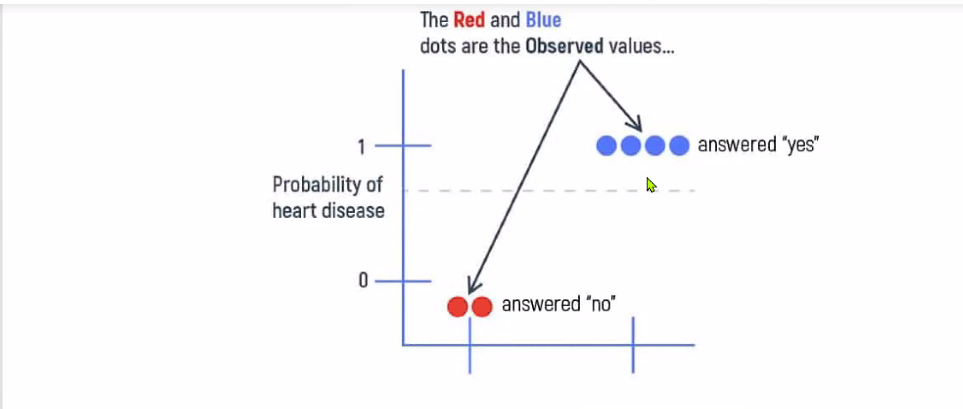
**ML\_07\_08\_2023\_Gradientboosting**

Gradient boosting’de olasiliklar uzerinden hesalama yailiyor.

amac residuallari minimize etmek



Butun olayimizi Olasiligi belirliyor hangi klass buyukse butun sinifi o olarak aliyor. Butun classi 1 olarak dusunuyor mesela. Aradaki farki aliyor.residual.



O yuzden 1 ile basliyor. O yuzden 0’in olasiligi ise -0.67 diyor mesela.

1’ler kalp krizi riski tasiyanlar.

Herbir gozlem icin baslangic olasiligini belirleyecek. Ona gore residuallari hesaplayacak.

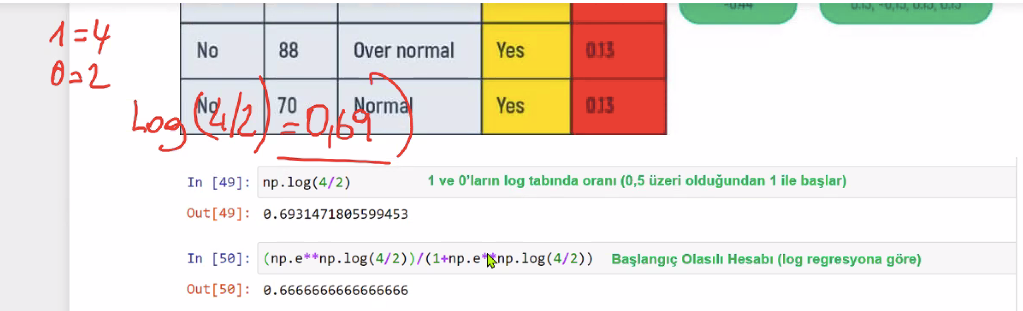
Threshold da hangisi yuksekse onu secerek basliyor.

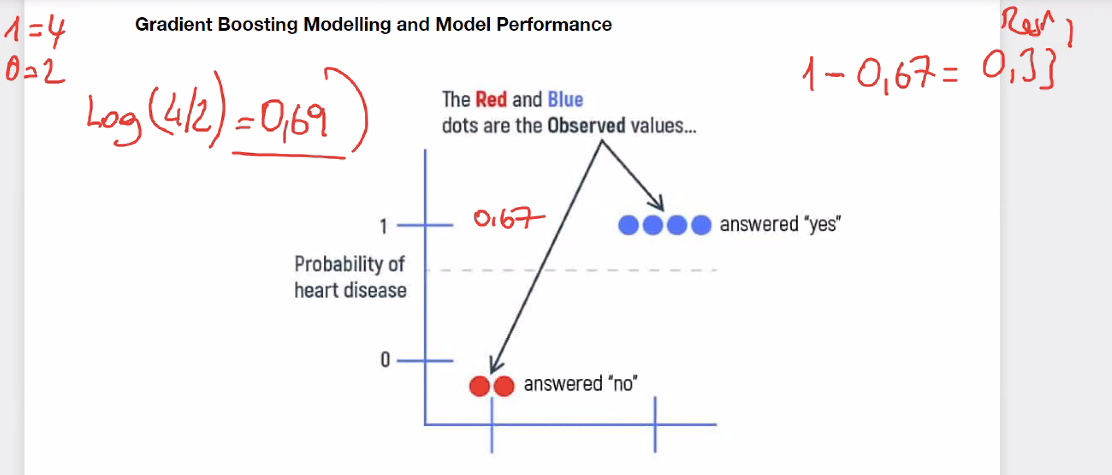
Ilk agac icin 0.67’den yola cikarak residuallarin olasiligini hesapliyor.

1’den 4 tane, 0’dan 2 tane gozlemimiz var. Bunun logoritmik tabaninda hesabini yapiyor. Bu benim baslangic olasiligim oluyor. Yani 0,67 benim baslangic olasiligim.

Hangisi 0.5’in uzerinde ise butun gozlemlere onu diyor. Burda 1. burdaki fark da residual.

Baslangic olasiligi icin loss function’a sokuyor:

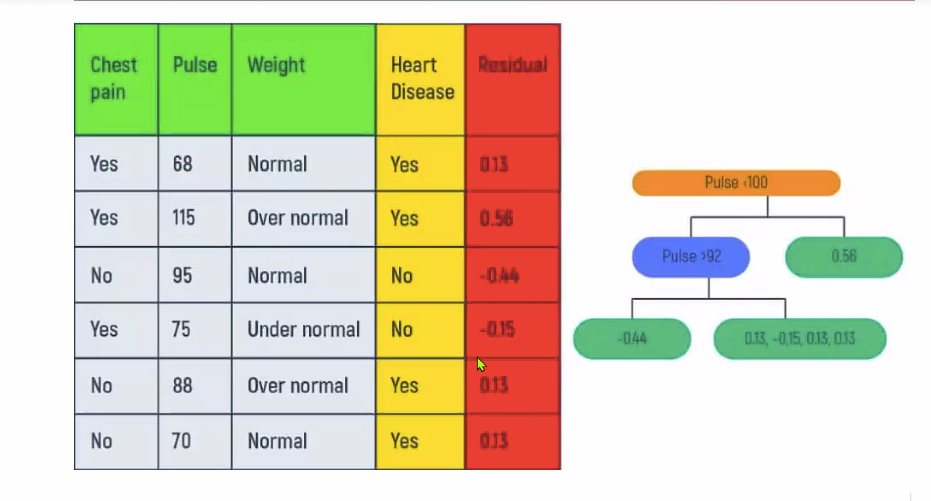




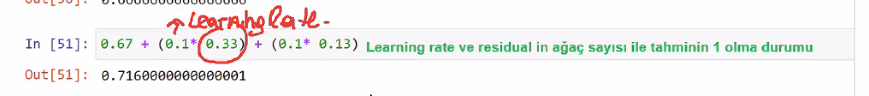
Yani benim ilk agacim icin 1-0,67=0,33 residualim.



1. Agac:



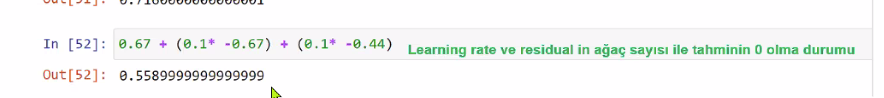
Iki agac sonrasi 1 olma durumum:



Baslangic olasiligi+learning rate\*residual+ 2.agac icin learning rate\*residual

Ilk residuali 1 icin 0,33 buldu. 0 icin de -0,67 cikti.

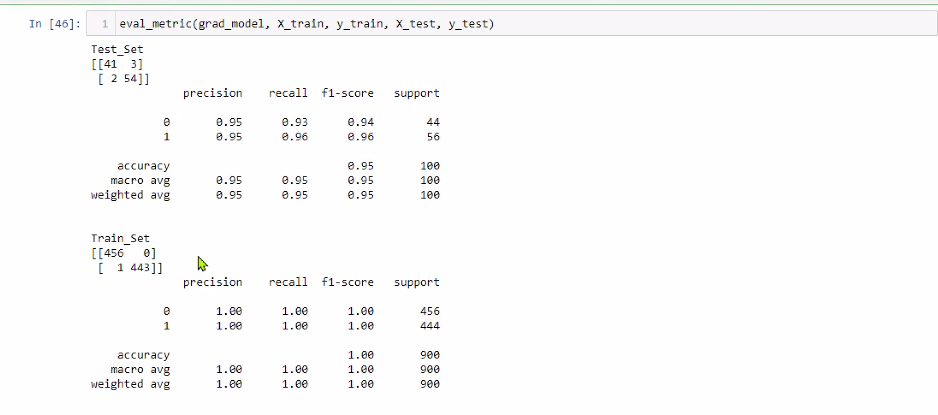
**2 agac sonrasi 0 olma olasiligi:**

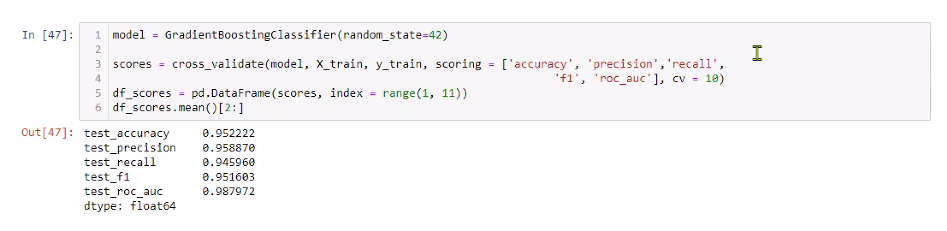


Agac sayisi arttikca 1 olanlar 1’e, 0 olanlar 0’a yaklasmali.



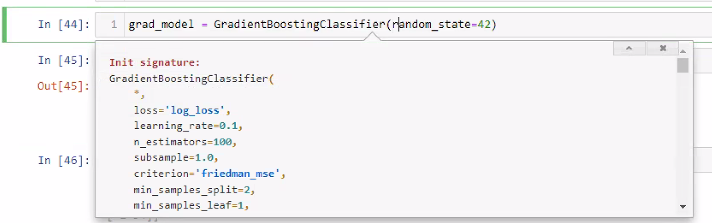
**Ilkel skor:**





CV bize datanin geneli ile ilgili bilgi veriyor. O yuzden overfit yok.

**Parameters in gradient boosting**



Loss-->logistic

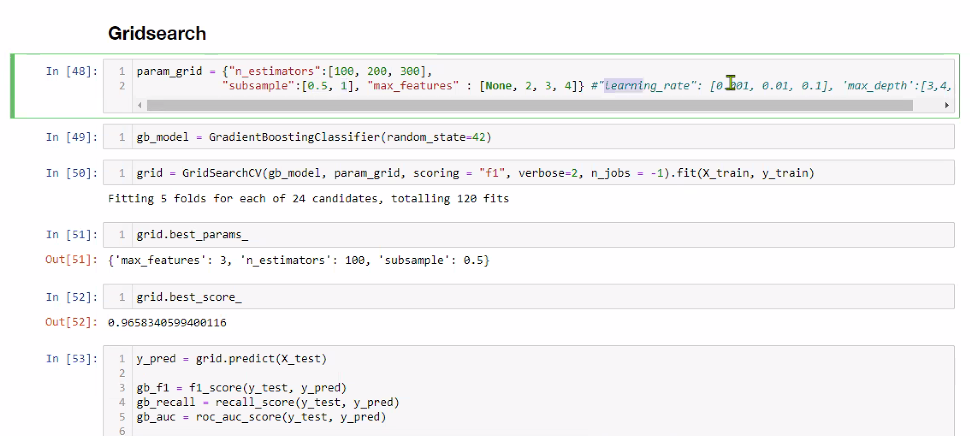
Learning\_rate-->yamactan inerken attigimiz adimlarin buyuklugu

Subsample--> 0.5 gibi, RF’deki datadaki gozlem sayisi belirlemek gibi estimatorlara gidecek gozlem sayisini ayarlayabiliyoruz.

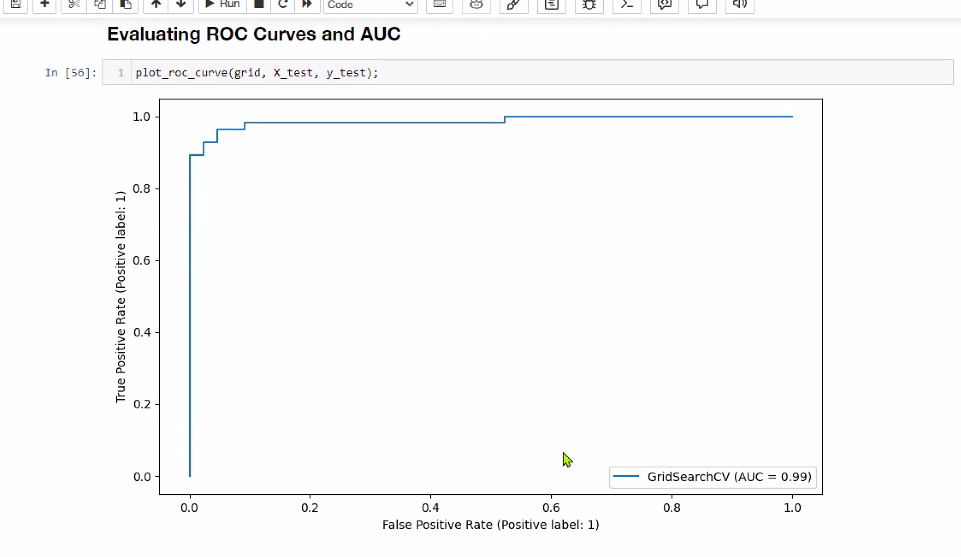
Criterion-->gini yerine bunu kullanin demek. Friedmanin gelistirdigi.

Max\_depth--> default 3, 2’yi de deneyin!

Max\_features-->default none, butun feature ile giriyor.



100, 500, 1000 verebilirsin mesela n\_estimator’a

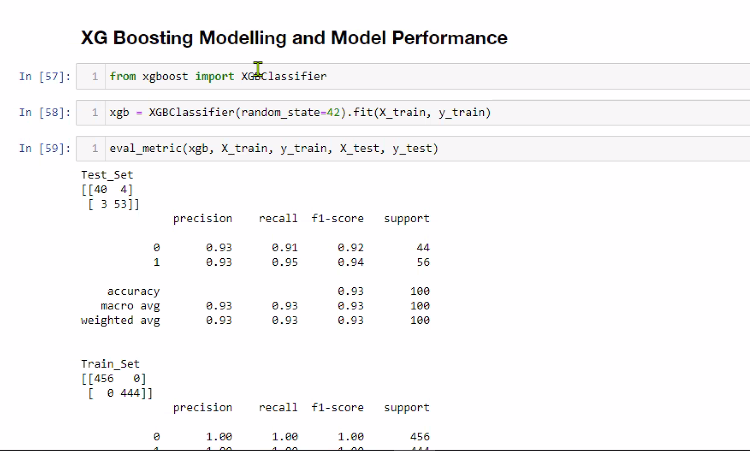


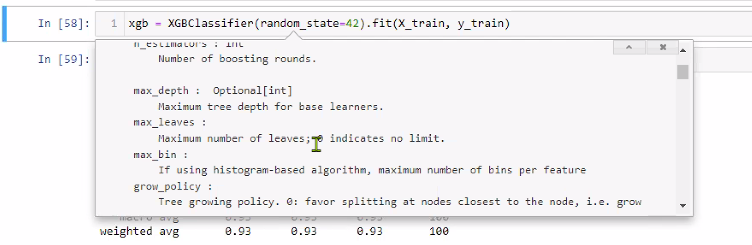
Tek farki logistic loss function, kullaniyor, residual hesapliyor.

**XG boosting**

Gradient boosting algoritmalarinin daha iyi optimize edilmis hali

Yani regularizasyon arametreleri ile oynanabiliyor.





Max\_bin--> histogram tabanli bir algoritma kullaniliyorsa max kutu sayisi

Verbosity--> detay, 0-->detaysiz, objectivity

Arka planda logistic regresyonu kullaniyor

Booster--> modelleri birlestiriyor, log ile lineari.

Gama-->ince ayar parametresi, min\_child\_weoght ise yaramazsa kullaniliabilir

Min\_child\_weight--> rakam buyudukce lasso gibi davranir. Overfit sorunu varsa ilk bunla oyna

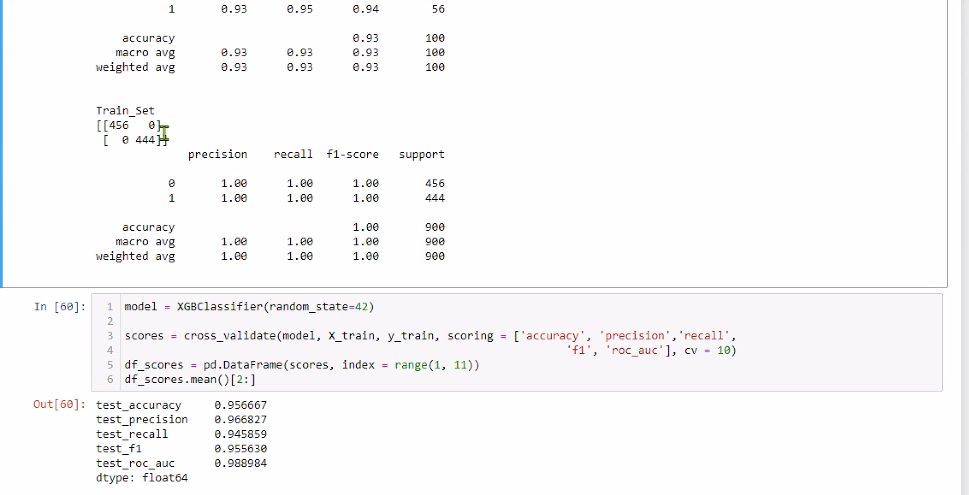
Subsample--> agaclarda estimatorin ne kadarini kullansin

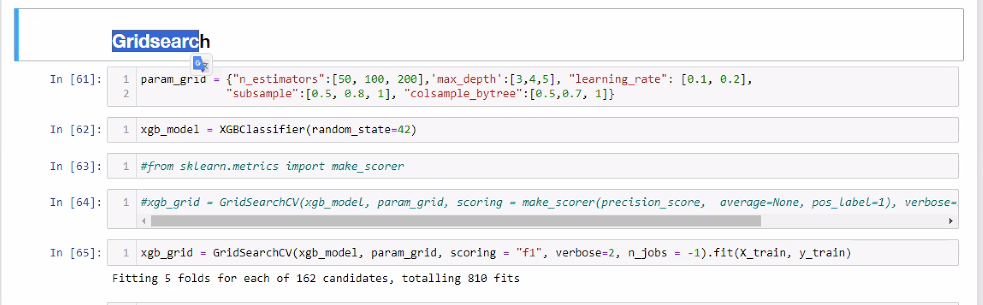
Colsample\_bynode --> bi nodde kalacak max yaprak sayisi

Scale\_pos\_weight--> imbalance bir durum varsa bunla cozebiliriz.

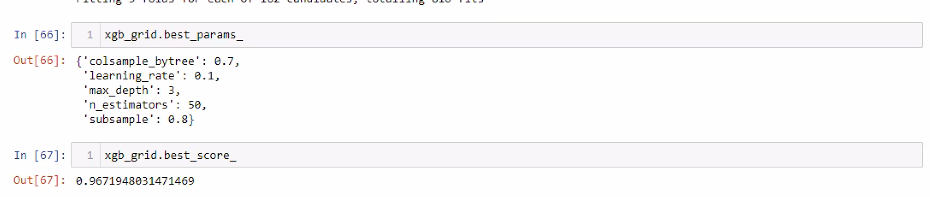
Buyuk olan rakami kucuge bol, 10 ve 8 vererek unbalanced durumunu balanced yapabiliyorsun.

|  |
| --- |
| Mustafa hocamin notu:  gradient boosting dahada optimize edilmiş hali bunun için regularizasyon parametreleriyle oynayabiliyoruz.  Baklım içine hangi parametreler var;  n\_estimators : int ağaç sayısı  max\_depth : Katman sayısı  learning rate :  verbosity : raporlama ile ilgili. model çalışırken arkada rapor yazsın mı yazmasın mı diye  objective : binary:logistic arka planda lojistik regressionu kullanıyor  booster : gbtree, gblinear burada modelleri birleştiriyor  reg\_alpha : L1 lasso Bu parametrelerle regularizasyon yapabiliyorum aynı zamand.Örnek veryorum lasso ile overfiti yok edicem o zaman buraya 1 alttakine 0 verirsem otomatikman lasso gibi davranır.  reg\_lambda : L2 Ridge  min\_child\_weight : rakam büyüdükçe lasso gibi davranır. eğer datmızda ciddi bir overfitting varsa ilk olarak bununla oynayabilirsiniz.  subsamples : Aynı her ağaçta gözlemimin ne kadarını kullansın  colsample\_bytree : Optional[float] burada max feature yok onun yerine buraya oran vererek maks featureyi ayarlayabiliyorsunuz  colsample\_bylevel : burada her bölünme için oran belirleyebiliyoruz ancak ubsample ve colsample kullanıldığında bunu kullanmanıza gerek yok. çok  colsample\_bynode : bir node ta kalacak maksimum yaprak sayısı gerek yok  scale\_pos\_weight :eğer imbalance durumu varsa 1=10, 0=100 100/10=10 çoğunluk class ın değerini azınlığın değerine bölüyoruz. değeri 10 ver, 8 ver oyna biraz |

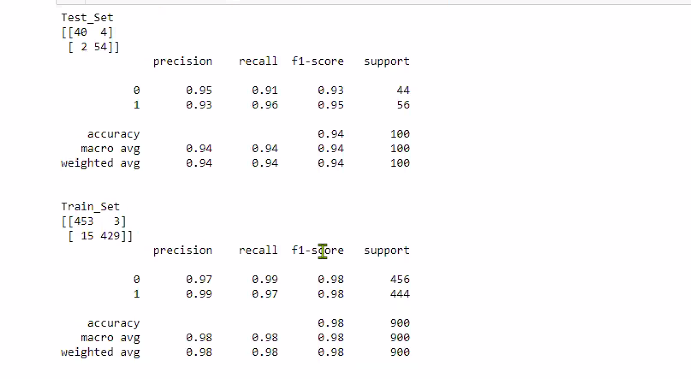




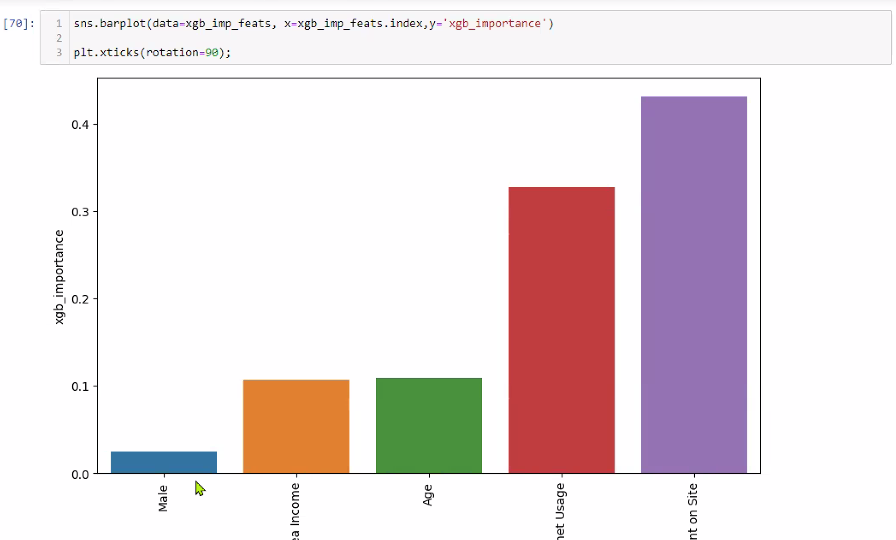
Modellerden en guclu yonlerini almis



Metriclerim

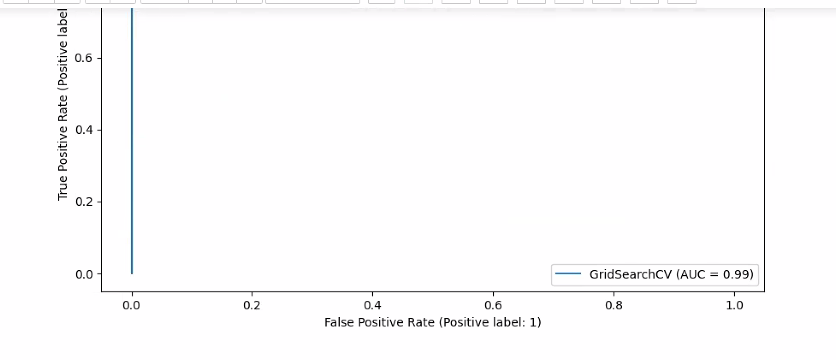


Skorlar dengeli



Male’den mesela daha cok insight almis

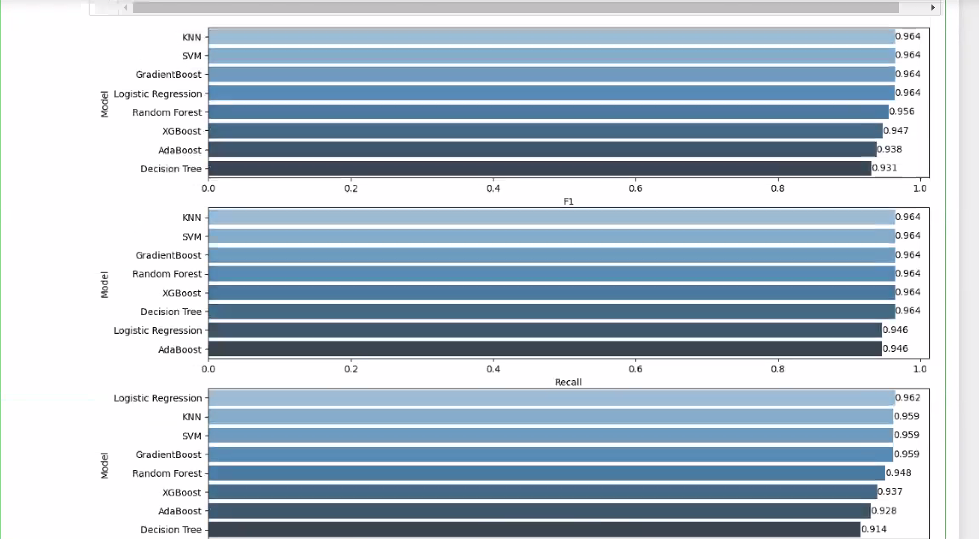
Xg boosting’in feature importance’i RF gibi kiymetli. Sorgusuz kullanabilirsin



**Model karsilastirmasina donuk ne yapabiliriz?**

RF, DT, Logistic, SVM, KNN, Hepsinin skorlarini aldim. Kiyaslayarak karar verecegiz.





KNN tree based--> supheyle yaklas.

SVM is maliyeti olan bir algoritma--> supheyle yaklas. Gecerliligi digerleri kadar degil.

Logistic regression is the best!

Icinden birkac algoritma secip hyperparametreleri ile oynayarak secim icin biraz daha hyperparameter tuning yaparsin.