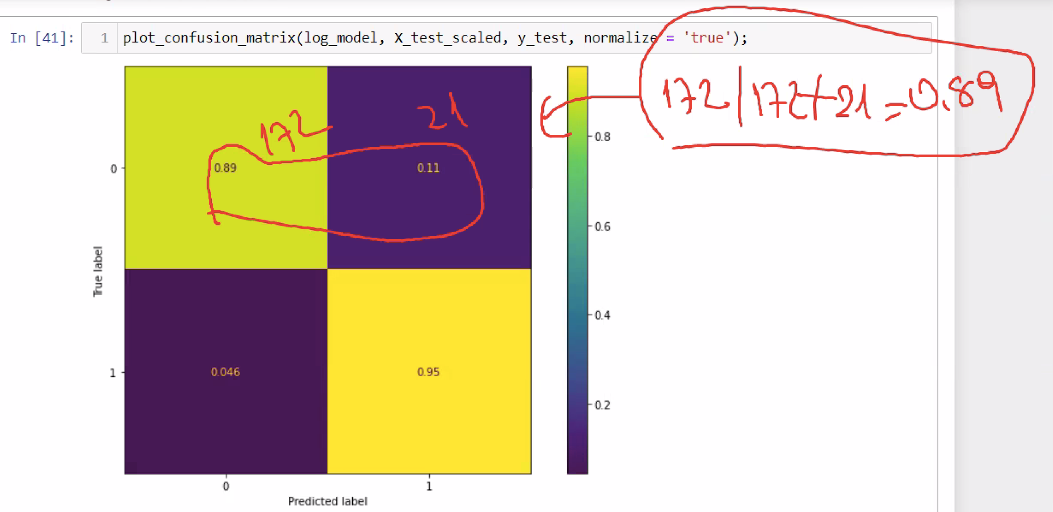
ML 20.07.2023



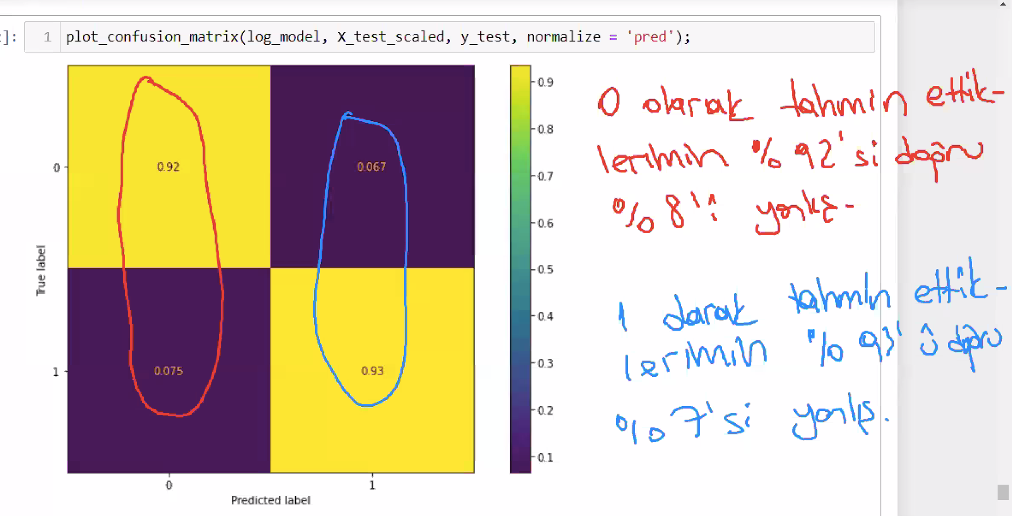
normalize='true' seçeneğini kullandığınızda, her satırın toplamı 1'e eşit olacak şekilde karışıklık matrisinin değerleri ölçeklendirilir. Bu, her bir gerçek sınıf için, modelin bu sınıfı doğru veya yanlış olarak ne sıklıkla tahmin ettiğini daha iyi anlamak için kullanışlıdır. Özellikle, sınıf dengesizliği olan durumlarda bu normalizasyon, her bir sınıfın performansını daha iyi değerlendirmenize olanak tanır.



100 olanlarin %95ini bilmis %5’ini bilememis

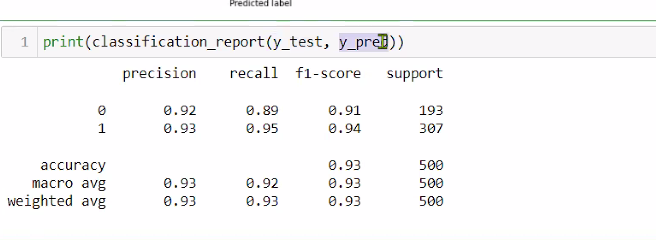
0 olanlarin ise %89’unu bilmis, %11’ini bilememis.

Tahminlerimin %92’si dogru.



0 olarak tahmin ettiklerimin %92’si dogru %8’i yanlis

1 olarak tahmin ettiklerimin %93’u dogru %7si yanlis



0 class:

Accuracy 0.93

Recall 0.89

Precision 0.92

1’e gore ise :

Accuracy 0.93

Recall 0.95

Precision 0.93

Sadece bir degere bakmak dogru degil.

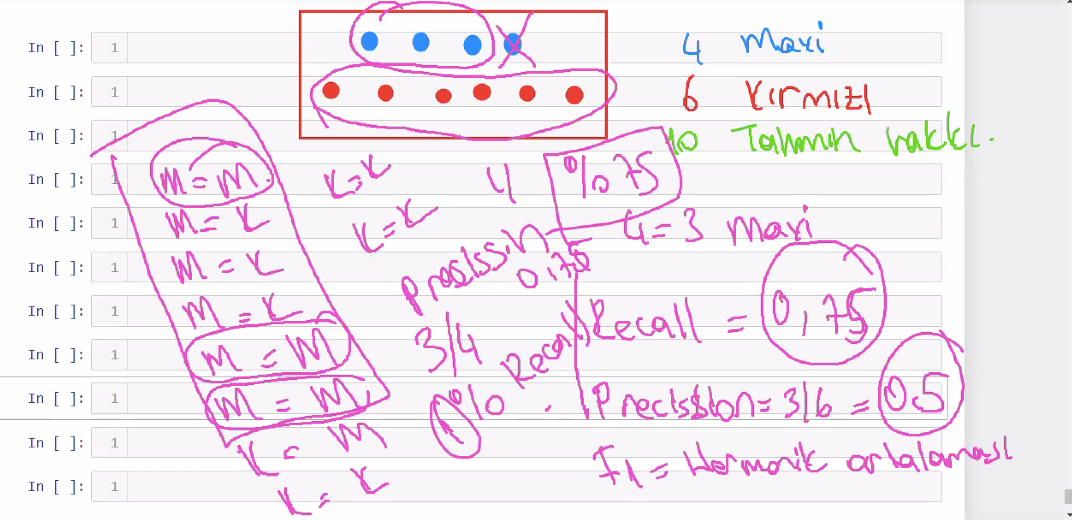
Ornegin accuracy.

Burada butun degerlerin birbiri ile uyumlu olmasi lazim

Burada bir uyum yakaladigimi dusunuyorum.







6 tahminden 3’unu tutturduysak precision 0.5 olur

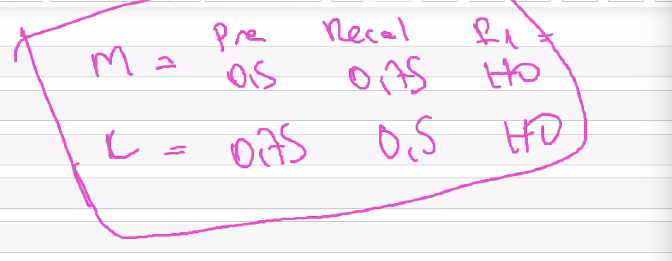
Harmonic ortalama

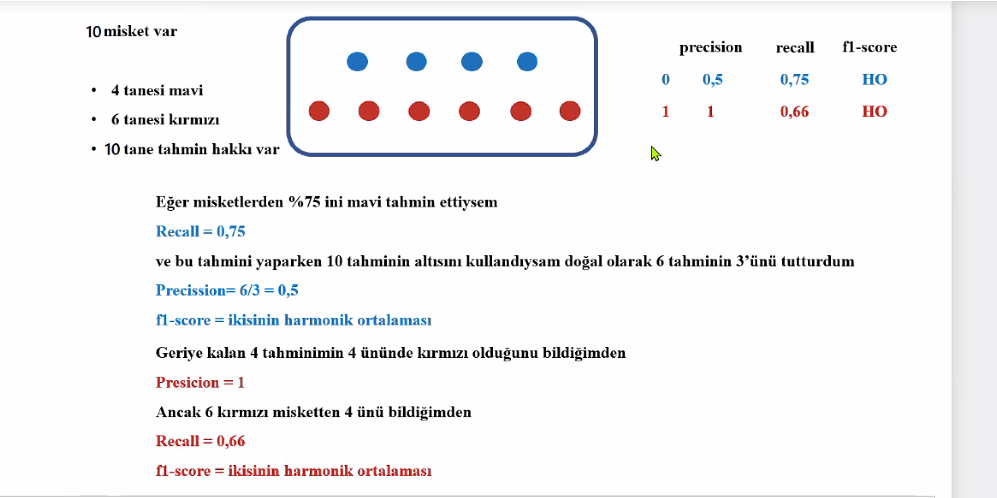
precision: toplam mavi tahminlerimden kac tanesini dogru bilmisim--> 3/6=0.5

recall: toplam gerceklesen mavilerden kac tanesini dogru bilmisim--> 3/4=0.75

Accuracy: toplam dogru bildiklerim (mavi(3)+kirmizi(3))/toplam tahmin-->6/10=0.6

|  |  |  |  |
| --- | --- | --- | --- |
|  | precision | recal | F1 |
| mavi | 0.5 | 0.75 | HO |
| kirmizi | 0.75 | 0.5 | HO |



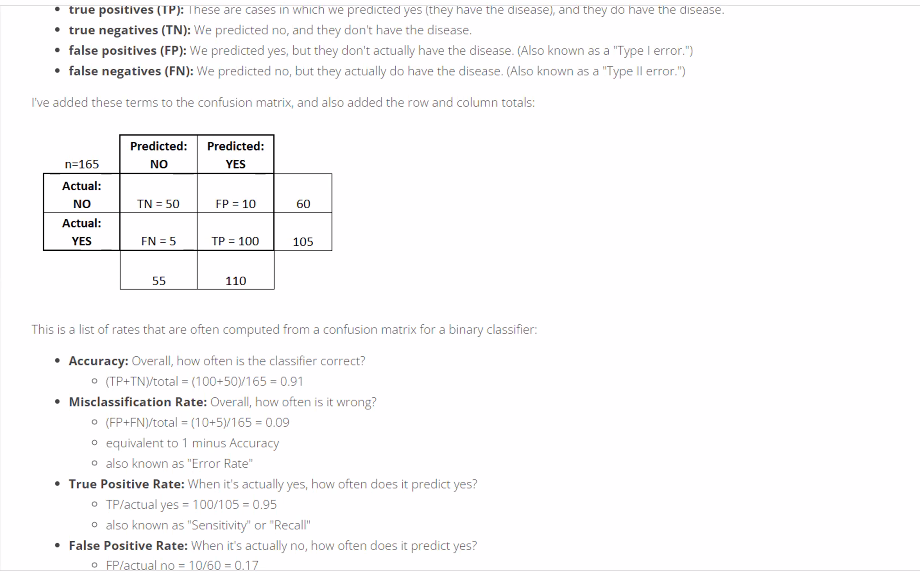


#10 misket olacak.

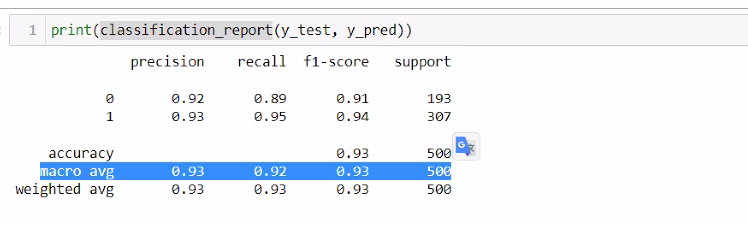
Target-1, Recall’um dusuk.

Data setinin tipine gore precision-recal onceligi degisebilir. Kanser vakasi mesela, recall’u yukseltebildigim kadar yukseltmeliyim.

Balance olmayan datalarda mesela accuracy yorumlamasi iyice yaniltici.



https://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/



0’la 1’in agirlikli ortalamasi--> **weigted avg**

Tum dogrularimin tahminlerime orani--> accuracy

Hedef klasimizin yuksekligi daha onemli. Hedef klasimiz icin en yuksek hangisi ise onu secebilirim. Burda 1-->isitme testini gecenler. En yuksek recall.

F1--> dengeyi gosterir.

F1’in yuksek olmasi, recallun yuksek olmasi otomatik olarak precision’in yuksek oldugnun gosterir.

classification\_report ile tüm skorlaı gösterir.

support = test datamın içinde kaç tane 0 kaç tane 1 değeri var.

macro avg = 0 ile 1'in ortalamsı

weighted avg = 0 ile 1'in ağırlıklı ortalamsı 500 ise bu oranların test datasının tamamına göre verildiğini gösterir.

accuracy = Benim tüm doğrularımın tahminlerime oranı

recall = burada hedef class a göre yorum yapalım bizimki 1 di. ben 1 olanların %'de kaçını bildim. biz 1 class ına göre baktık

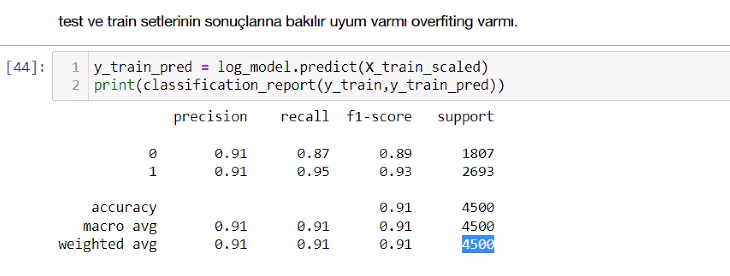
precision = Benim 1 dediklerimin % kaçı doğru. tahim edilenlerde kaçı tanesi doğru. tahmin edilenden kaç tanesini bilmiş bunun oranı. Burada anlaşılması gereken husus ben tahmin yaparken belki 100 doğru değer için 105 tahminde bulundum doğal olarak bu değer azıcık düşmüş. Kısaca modelin tahmin gücünü gösteriyor diyebiliriz. ne kadar çok tahmin yaparsa değer düşer. Kısaca ben 193 tane hasta deyim hastayı tahmin ederken 500 tahminde bulunsun istemiyorum bu sebeple recall ile precission arasında bir denge olmalı.

f1 = precission ve recall arasındaki dengeyi sağlar.

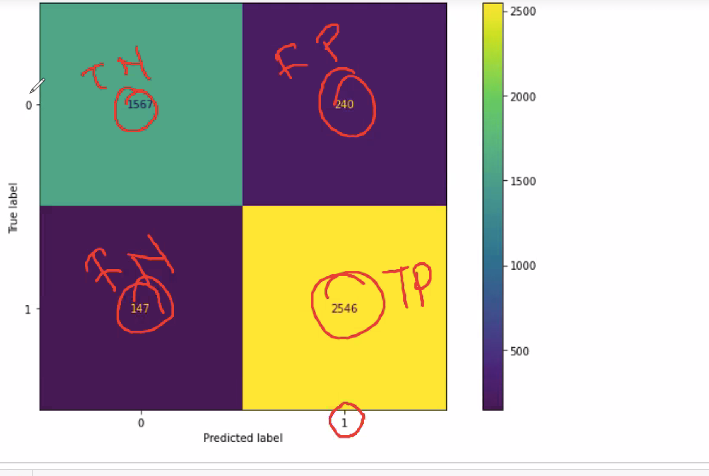
precision ve recal değerleri dengede olmalı birbirine yakın olmalıdır.

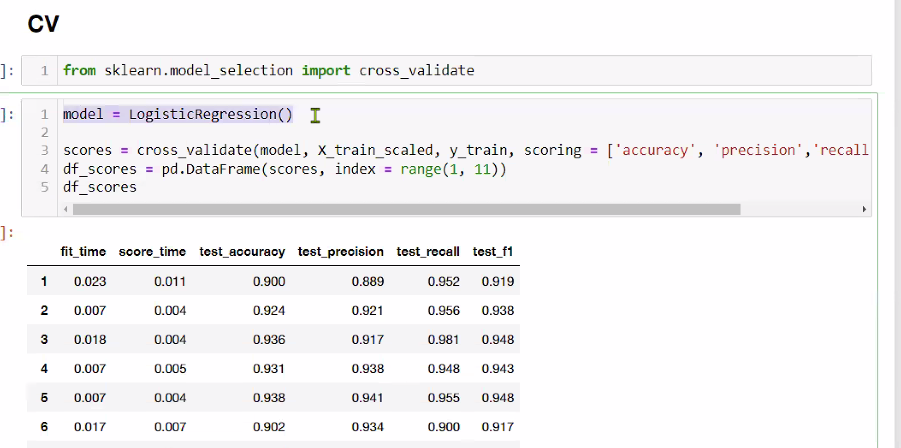
aşağıdaki sonuçta 1 clasının skorları biribriyle uyumludur.

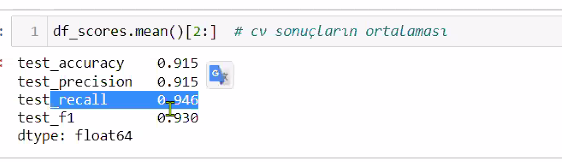
test ve train setlerinin sonuçlarına bakılır uyum varmı overfiting varmı.



Burda f1 bizim icin onemli. Ikisini de birlestiriyor







Butun skorlar birbiri ile uyumlu o yuzden overfittingden bahsedemeyiz.

**Test ve train datasini alt alta gormek icin yazacagimiz formul:**

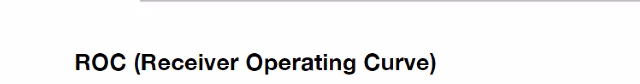
print("Test Set")

print(classification\_report(y\_test,y\_pred))

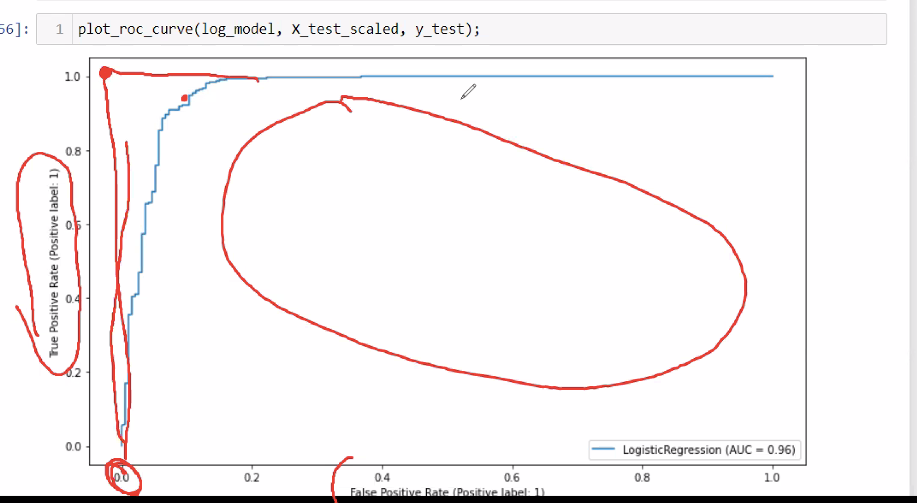
print("Train Set\n")

y\_train\_pred = log\_model.predict(X\_train\_scaled)

print(classification\_report(y\_train,y\_train\_pred))



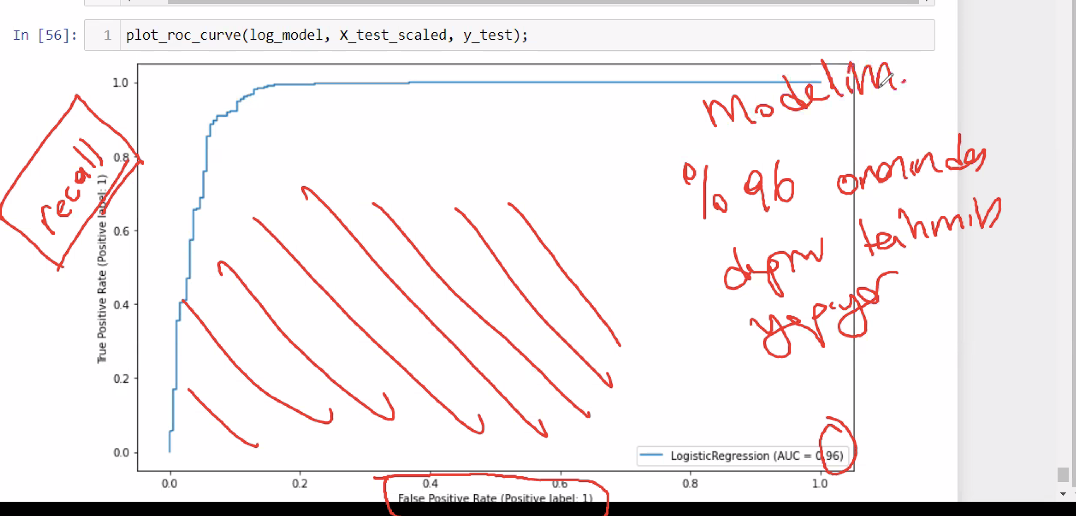
Performansi degerlendirmek icin kullaniliyor

#

x--> false positive

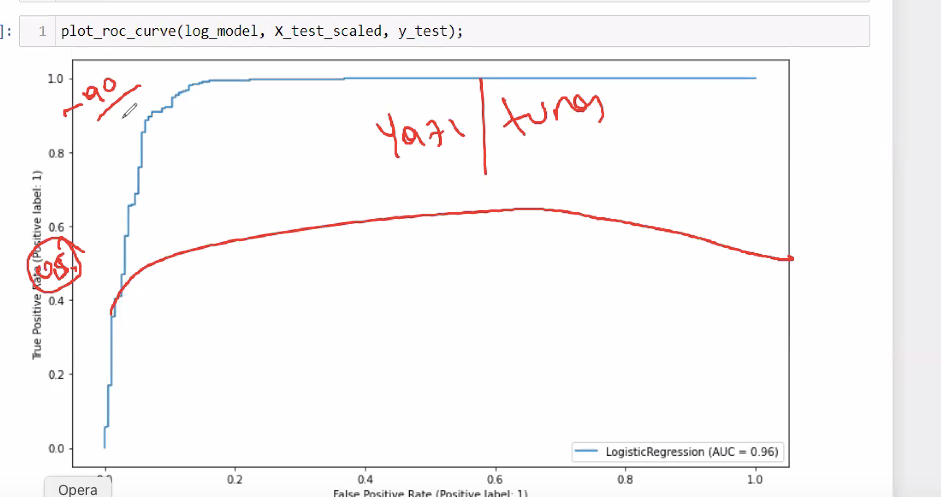
Y--> true positive, 1 olanlarin %kacini bildim dedigim yer, recall

Cizgi ne kadar koseye yakinsa modelimin genelleme yapma yetenegi o kadar yuksek demektir. Y



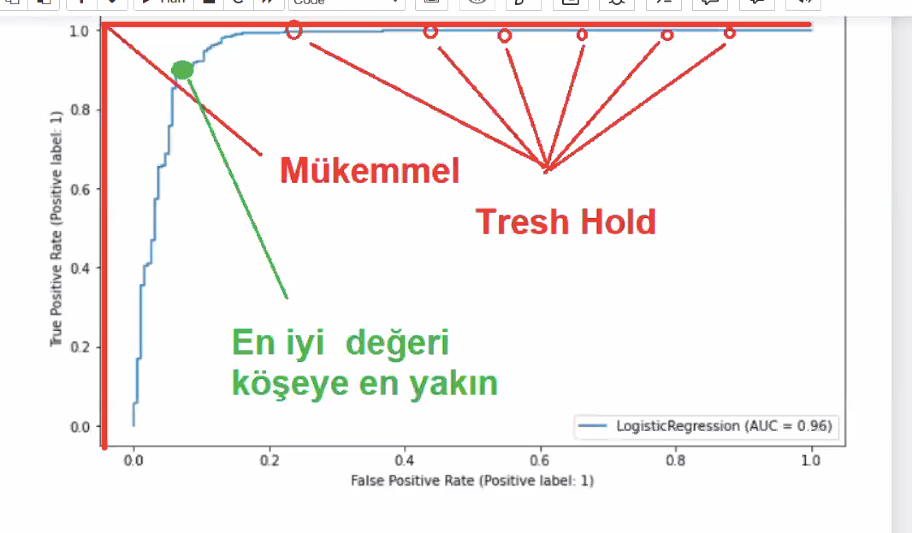
Modelim %96 oraninda dogru tahmin ediyor demektir. Alan ne kadar buyukse modelim o kadar dogru tahmin ediyor demektir.

Literatur %90 uzerini cok basarili kabul ediyor.

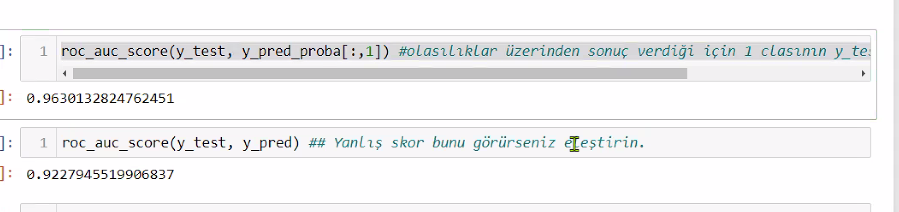


Egri 0.5’lere iniyorsa, yazi tura tipi tahminde bulunuyor demektir.

Amacim modelimin %90’larin uzerinde tahminde bulunmasi yani cizginin yukarilarda olmasi. Yani alanin buyuk olmasi.



|  |
| --- |
| grafiğin y ekseni yani True Possitive rate = recall yani gerçekte 1 olanların yüzde kaçını bildim.  x ekseni ise False possitive rate bunun başka bir adı yok:) buda sıfırların %'de kaçını bilemedim.  Bu grafiğin alanı ne kadar büyükse tahminim o kadar büyük olasılıkla doğru tahmin yapıyor demek. Yani %96 oranında doğru tahmin yapıyor.  burda verdiği değerin yüksek olması modelimin çok iyi bir ayrıştırma yapması demek dost ile düşmanı ayırt ediyor.  literatür %90 ve üzerine çok başarılı kabul ediyor.  doğal olarak asadece recall f1 accuracy precission değerlerine bakmayacağız modelin genel performansını değerlendirirken bu skora da bakacağız. |



Burada neden fark çıktı peki

çünkü buradaki değerler 0 ve 1 e göre

ama proba daki değerler olasılık

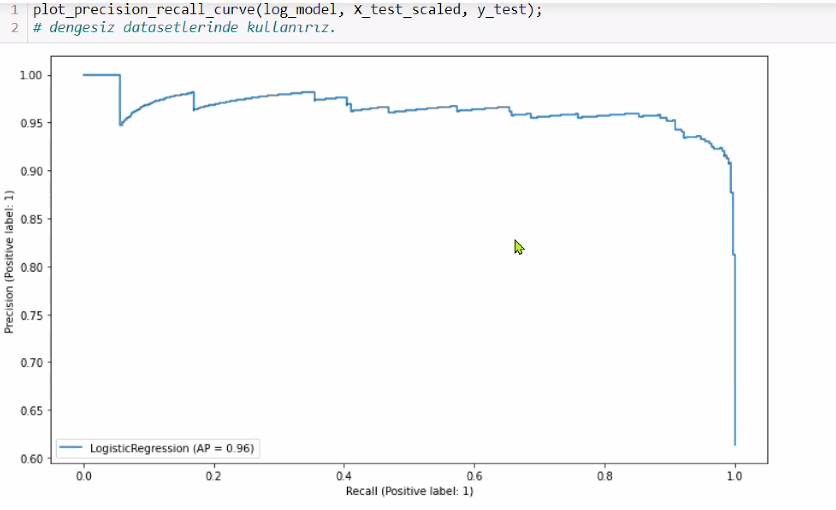
şöyleki olasılık yaparken 0,45 de 0 a geçti 0,01 de değilmi

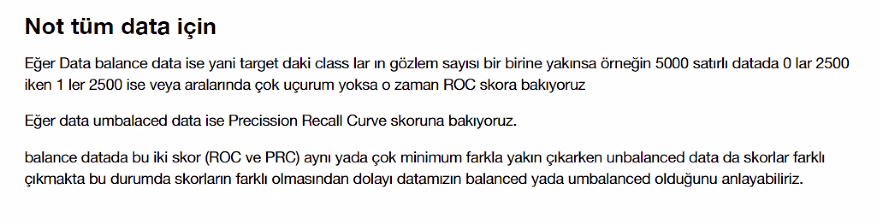
işte olasılık değerleri direk bu değerler olduğundan bu sebeple proba daha yüksek çıktı. ROC AUC curve bunula hesaplanıyor

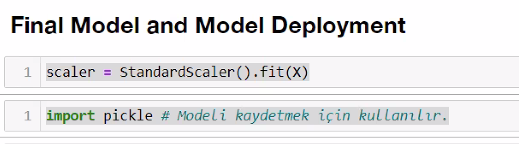
balanced bir datam varsa, roc’a bak

Balance data degilse, precision\_recall\_curve’e bak. Zaten de farkli cikar.

roc ile precision\_recall\_curve ayni ise, zaten bunlardan guzel sonuclar cikar.







Modelimde scale yaptiysam scaler diye kaydediyorum

Modeli kaydetmek

Bir tahmin modeli kurduk, hedef class’a gore cok yuksek tahmin orani olan bir model bu.

Bunu kaydedecegiz simdi

Model deploya bir hazirlik bu. Bir onceki asamasi.

