CLASSIFICATION BÖLÜM SONU

--> Biz biliyoruzki özellikle veri dengeli dağılmamışsa,

--> Yani training set'in %90'ı bir class %10'u diğer class şeklindeyse,

--> Accuracy bize çok da mantıklı bir evaluation sağlamaz.

--> Çünkü ben giderim hepsine YES derim %90 accuracy elde ederim.

--> Bu algoritmaya ZeroR algoritması deniyor.

--> Böyle durumlarda başka evaluation criteria'lar önem teşkil eder.

--> Bizim algoritmamızın değerli olması için zeroR'ı zaten geçmesi lazım, bunu göz önünde tutmalıyız.

--> Ayrıca precision, recall, F1 Score gibi yeni kavramlar ortaya çıkar.

""" ROC Eğrisi : Sınıflandırma Algoritmalarının Nasıl Karışılaştıracağı Hakkında Önemli bir Konu

--> ROC: Receiver Operating Characteristic (ROC)

--> ROC'u anlamak için true positive rate ve false positive rate iyi anlaşılmalı:

n=165	**Predicted: NO**	**Predicted: YES**	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

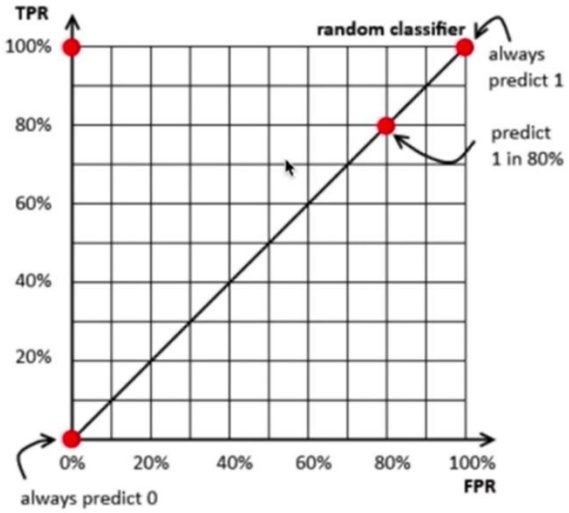
$$tpr = \frac{TP}{TP + FN}$$

$$fpr = \frac{FP}{FP + TN}$$

*tpr = TP / (TP+FN) : True olanlar arasından true olarak sınıflandırılanlar. Yüksek olması iyi.

*fpr = FP / (FP+TN) : False olanlar arasında true olarak sınıflandırılanlar. Yüksek olması kötü.

--> ROC çizmek için bir eksene tpr, bir eksene fpr konulur.



- ➔ 45 derecelik doğrunun üzeri random classifier'ı temsil eder yanı en kötü classifier'ların vereceği performansı gösterir. Bunun altında olanlar'ın zaten etiketini değiştirirsek yine onun üstüne çıkar!
- ➔ Rastgele sınıflandırmaların bu çizginin üzerinde olmasını bekleriz.

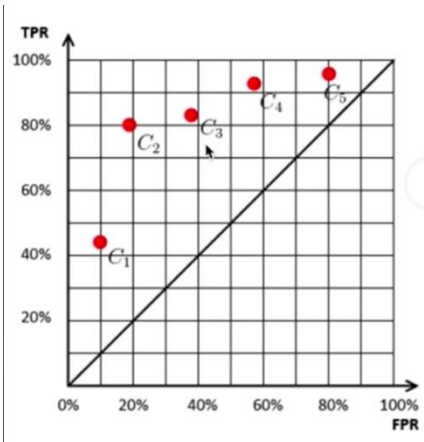
*tpr=1, fpr=0 ideal durumdur hem true hem false örnekleri iyi sınıflandırır.

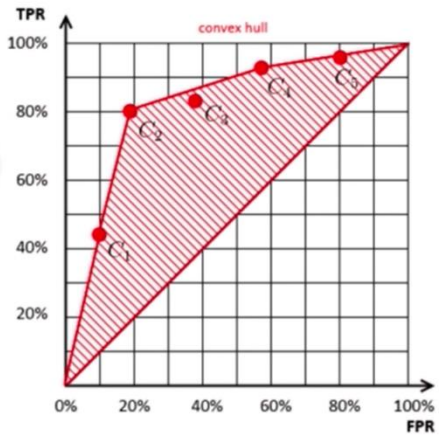
*tpr=1, fpr=1 ise true olanları %100 ayırıyor, false olanları hep hatalı ayırıyor demek.

--> Farklı algoritmaların performanslarını ROC üzerine yerleştiririz.

--> Daha sonra bu grafiğe göre seçim yapabiliriz

--> Tüm uç nokaları birleştirip bir çokgen oluşturursak, bu çokgenin içinde kalan noktalara denk gelen algoritmalar işe yaramaz.



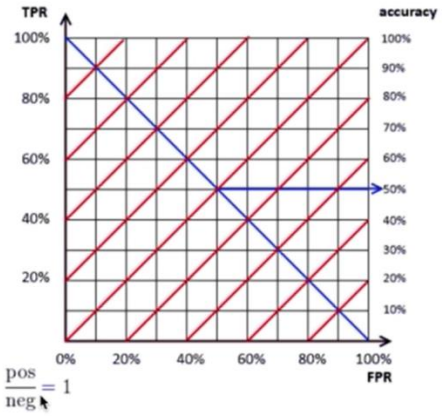


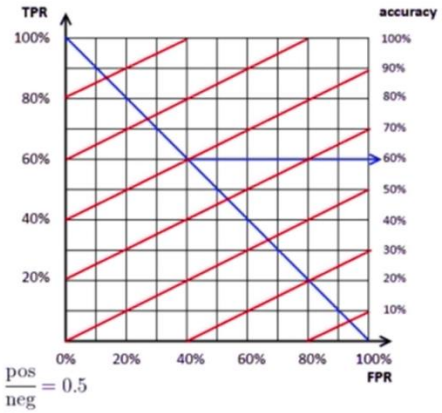
--> Çünkü içeride kalan algoritmaların daha iyileri var demektir. Çokgenin köşeleri!

→ Mesela C3, C2 ve C4 tarafından domine edilir.

--> Hangisini seçeceğim ayrıca prevalance değerine bağlıdır. Yani training set'in dağılımına bağlı.

Accuracy Doğruları





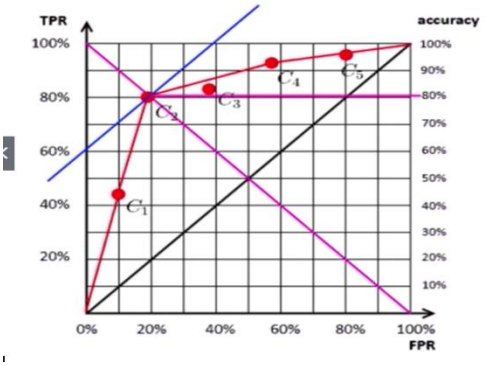
➔ İki class 50 - %50 'mi dağılmış yoksa birinden çok az örnek diğerinden çok fazla mı var bu önemli.

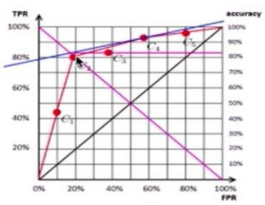
➔ Meslea pos ve negative dağılımı dengeli ise accuracy doğruları şekil bir gibi eğim 1 olacak şekilde yerleşir yani bu doğru üzerinde her noktanın accuracy'si aynıdır.

➔ Pos neg oranı 0.5 ise şekil 2 gibi bir dağılım olacak.

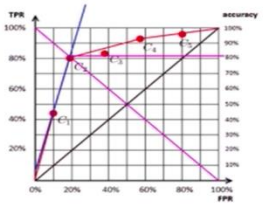
➔ Algoritma seçiminde bu dağımlar önemlidir.
➔ Mesela dağılım oranı dengeli pos/neg = 1 veya prevalence = 0.5

Algoritma Seçimi





neg/pos = 1/4



neg/pos = 4/1

➔ Göründüğü gibi dağılım oranına göre mavi çizgi ile pembenin kesişimi bulunur vee n yüksek accuracy odur. Dağılım oranına göre seçilecek algoritma değişir.

➔ AMA SONUÇTA ACCURACY'E BAKMIŞ OLUYORUZ BURADA!!! YANİ HİÇ ROC'A BAKMADAN DA ACCURACY'E BAKABİLİRDİK. BURADA ROC'UN AVANTAJINI ANLAYAMADIM TAM OLARAK.

➔ Yani ROC ile önce bazı algoritmaları direkt eleyebiliriz. (Area içinde kalanlar)

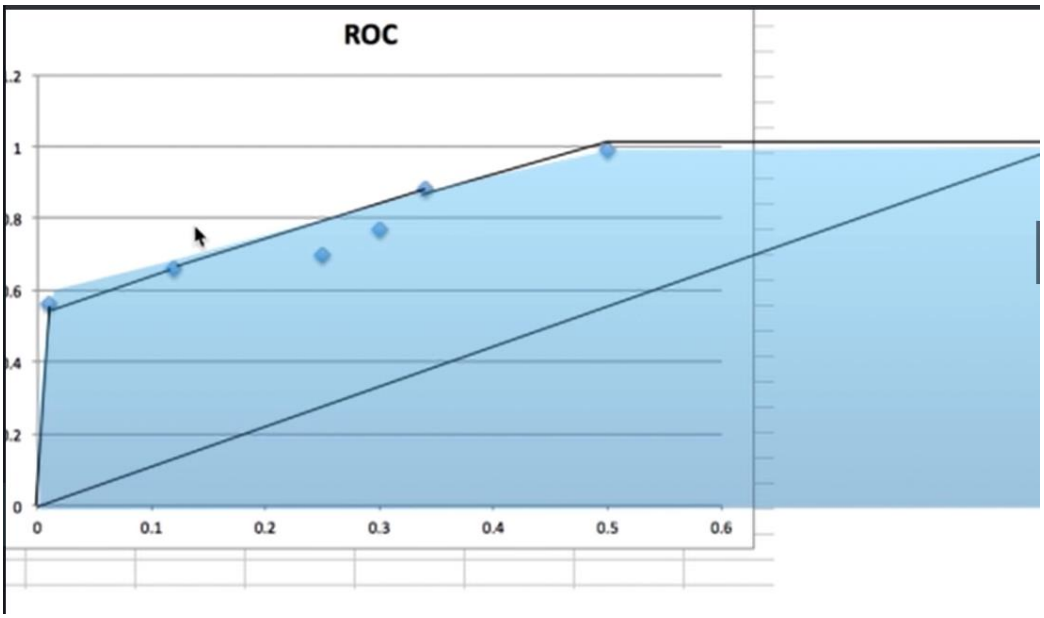
➔ Dağılıma göre hangisini seçmemiz gerektiğini söylüyor.

➔ Ayrıca hangi algoritmanın hangi dağılımlarda daha başarılı olacağını da bize söylüyor.

➔ Meslela neg/pos = 1/4 oranı için C5 en iyi accuracy'I veriyor ama ben grafik sayesinde şunu biliyorum ki eğer very dağılım dengesi değişirse C5 işe yaramayabilir. Veri dengesiz dağıldığı için C5 iyi performans gösteriyor.

AUC: Area Under Curve

→ Bunun olayı şu, ben training set'imin %10'u %20'si %30 u … gibi farklı boyutlarda ROC çizerim ve performansa bakarım. Eğer ROC performansı hızlı artarsa bu iyi bir algoritma yanı çabuk öğrenme sağlıyor diyebiliriz. Bunun ölçümünün bir başka yolu da alan hesaplama. Alan 1' e yaklaştıkça daha hızlı öğreniyor demektir. İdeal durumda alan 1 olur.



Sınıflandırma Algoritmaları Karşılaştırma ve Özet

Bölüm 3'te 7 sınıflandırma modelini öğrendiniz. Daha önce, 2. Bölüm'de Regresyon için sorduğunuz aşağıdaki soruları bu bölüm sonunda da soruyor olabilirsiniz:

1. Her modelin artıları ve eksileri nelerdir?
2. Problemim için hangi modeli seçeceğimi nasıl bilebilirim?
3. Bu modellerin her birini nasıl geliştirebilirim?

Yeniden bu soruların her birine tek tek cevap verelim:

1. Her modelin artıları ve eksileri nelerdir? Burada her sınıflandırma modelinin tüm artılarını ve eksilerini veren bir özet sayfası bulabilirsiniz (buraya tıklayınız).

2. Problemim için hangi modeli seçeceğimi nasıl bilebilirim? Regresyon modelleri için olduğu gibi, probleminizin doğrusal mı yoksa doğrusal olmayan özellikte mi olduğunu anlamanız gerekir. Bunu Bölüm 10 - Model Seçimi dersinde öğreneceksiniz. Sonra: Eğer probleminiz doğrusal ise, Lojistik Regresyon veya SVM'yi kullanabilirsiniz. Eğer probleminiz lineer değil ise, K-NN, Naif Bayes, Karar Ağacı veya Rastgele Ormanı'nı kullanabilirsiniz. Peki hangisini her durumda seçmeliyim? Bunu Bölüm 10'da Model Seçimi k-Katlama Çapraz Doğrulama ile daha detaylı anlatıyor olacağız.

Algoritmalara farklı problemler açısından bakıldığında şunları kullanmayı tercih ederbilirsiniz: - Tahminlerinizi olasılıklarına göre sıralamak istediğinizde Lojistik Regresyon veya Naive Bayes. Örneğin, müşterilerinizi belirli bir ürünü satın alma olasılığı en yüksek olanından en düşük olanına doğru sıralamak istiyorsanız. Bu sayede, pazarlama kampanyalarınızı hedeflenmiş müşterilere yönlendirebilirsiniz. Ve tabi ki bu tür bir iş problemi için, eğer probleminiz doğrusal ise Lojistik Regresyonunu, ve probleminiz doğrusal değilse Naive Bayes'i kullanabilirsiniz. - Müşterilerinizin hangi segmente ait olduğunu tahmin etmek istediğinizde SVM. Segmentler, örneğin, her türlü segment olabilir. Örneğin daha önce kümeleme ile tanımladığınız bazı pazar segmentleri. - Model sonuçlarınızı net bir şekilde yorumlamak istediğinizde Karar Ağacı - Rastgele Orman gibi yöntemleri yine örneğin sadece daha az yorumlama ihtiyacına sahip yüksek performanslı algoritmalar arıyorsanız kullanabilirsiniz.

3. Bu modellerin her birini nasıl geliştirebilirim? 2. Bölüm'deki ile aynı cevabı vermemiz gerekecek: Bölüm 10 - Model Seçiminde, ayarlarınızı yaparak, modelinizin performansını iyileştirmenize bakacağız, yine bu bölümün altında, Parametre Ayarlamaya ayrılmış ikinci bölümü bulacaksınız. Büyük olasılıkla zaten her modelin iki tür parametreden oluştuğunu fark etmişsinizdir: öğrenilen parametreler, örneğin Doğrusal Regresyondaki katsayılar ve hiperparametreler. Hiperparametreler öğrenilmeyen ve model denklemlerde sabit değerleri olan parametrelerdir. Örneğin, lambda regülasyon parametresi veya penaltı parametresi C hiperparametrelerdir. Şimdiye kadar bu hiperparametrelerin varsayılan değerini kullandık ve modeliniz daha yüksek performans göstersin diye bunların optimum değerleri için araştırma yapmadık. En uygun değerlerini bulmak, Parametre Ayarının tam olarak ne olduğuyla ilgilidir. Bu nedenle, model performansınızı geliştirmek ve bazı parametreler ayarlamayı yapmakla ilgilenenler için, doğrudan Bölüm 10 - Model Seçimine atlamak iyi bir fikir olabilir.

Şimdi Bölüm 3'ü tamamladığınız için tebrikler ve yolculuğun bir sonraki bölümüne geçelim: