GİRİŞ

Eğitmen

Özgür YILDIRIM

linkedin.com/in/Ozgur-yldrm/

github.com/OzgurYldrm

Sunum + Notebook + Homework → Github

- **Değerlendirme Metrikleri**
- **Veri Bölme Yöntemleri**
- **Çoklu Etiketli Sınıflandırma**
- **Güven Aralıkları**
- **Genelleme ve Aşırı/Ufak Uydurma**
- **Model Seçim Teknikleri**

Accuracy

Gerçek Değer	Tahmin
1	1
1	0
0	1

Accuracy = ⅓ = **%33**

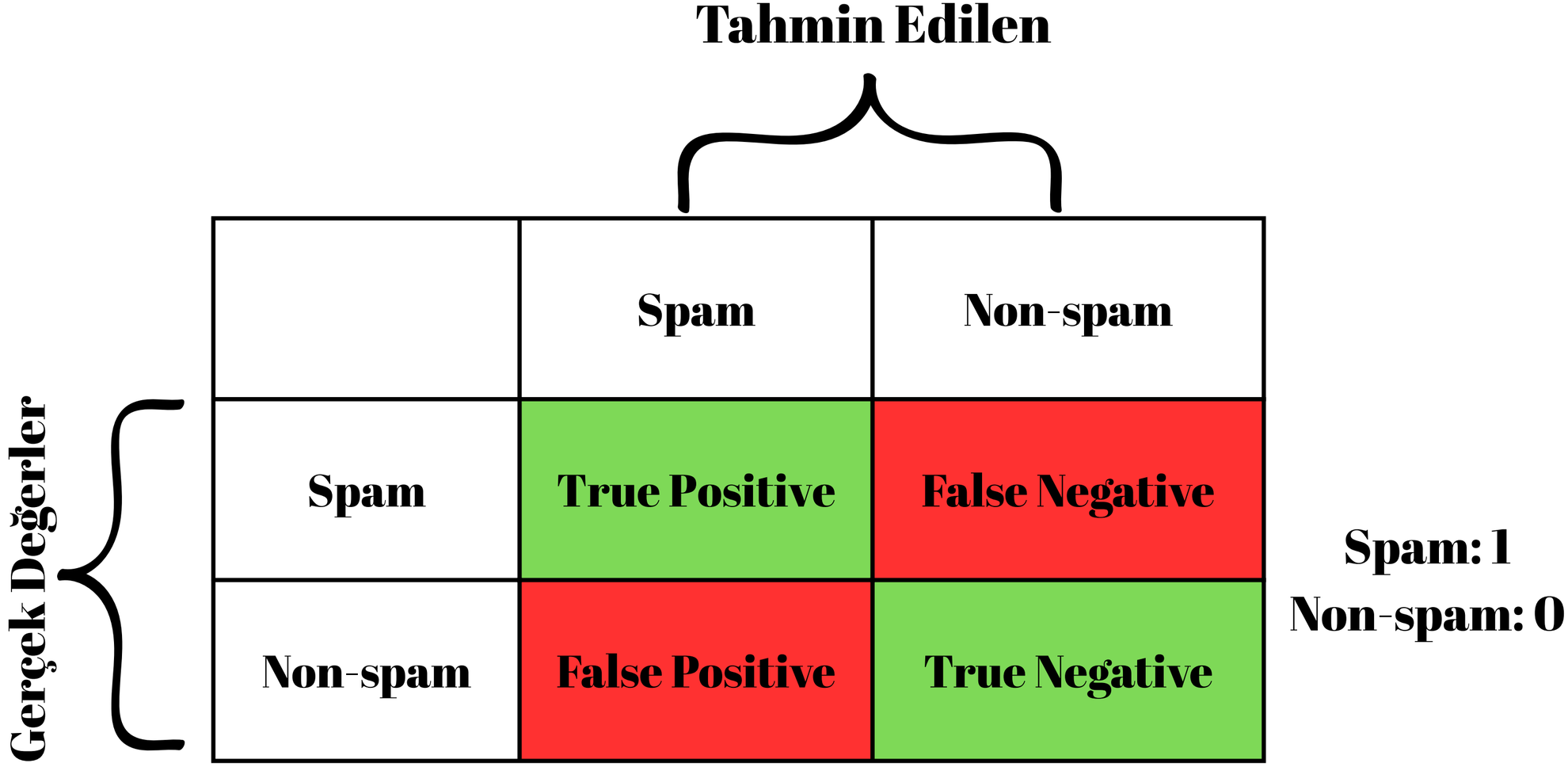
$$\text{Accuracy} = \% \frac{\text{Doğru Sayısı}}{\text{Tüm Örnek Sayı}} \times 100$$

- **En temel metrictir.**
- **Her zaman mantıksal bir sonuç vermeyebilir.**

Pozitif sayısı: 1
Negatif Sayısı: 99
Toplam veri: 100
Böyle bir durumda sadece negatif tahmin ederek %99 doğruluk elde ettik.

} **Daha bilgilendirici metric lazım**

Confussion Matrix



*Contingency Table

Birçok metrik için temel oluşturan bir tablodur.

Precision

	Spam	Non-spam
Spam	97	1
Non-spam	1	1

Precision = 97/98

	Spam	Non-spam
Spam	1	1
Non-spam	1	97

Precision = 1/2

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

Tahmin edilen olumlu (pozitif) örneklerin ne kadarı gerçekten olumlu?

Recall

	Spam	Non-spam
Spam	97	1
Non-spam	1	1

Recall = 97/98

	Spam	Non-spam
Spam	1	1
Non-spam	1	97

Recall = 1/2

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

Gerçek olumlu (pozitif) örneklerin ne kadarını doğru tahmin ettik?

Precision-Recall Tradeoff

Sınıflandırma problemlerinde kullanılan eşik değer genellikle 0.5 'tir. Yani 0.5 üzerindeki çıktılar pozitif (1) olarak etiketlenirken 0.5 değerinin altındaki çıktılar negatif (0) olarak etiketlenir

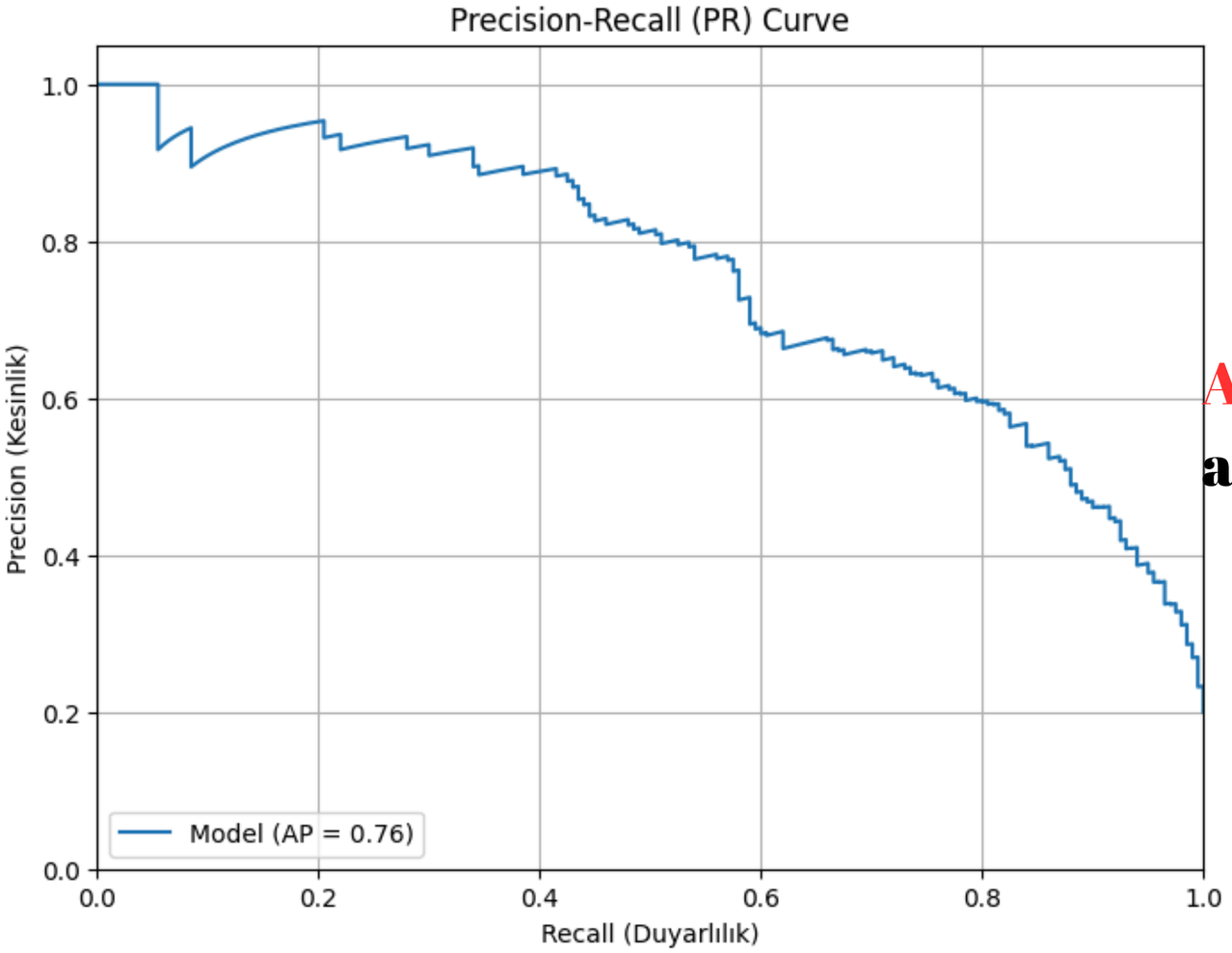
- Eşik değer aşağıya çekilirse (0.3 gibi) , daha fazla örnek pozitif olarak etiketlenir ve recall artar.
- Eşik değer yukarıya çekilirse (0.7 gibi) , daha az örnek pozitif olarak etiketlenir ve precision artar.

Eşik değerin kaç olarak belirleneceği tamamen çalışmaya bağlıdır.

- Kanser teşhisi yapan bir modelin var → düşük eşik değer, yüksek recall istersin (pozitifi tespit etmek negatifi kaçırmaktan daha önemli)
- Spam tespiti yapan bir modelin var → yüksek eşik değer, yüksek precision istersin (kesinlikle pozitif (spam) olmasını istersin yoksa yanlış mail spam'e gidebilir)

Imbalanced Dataset → Sınıfların dağılış oranının dengesiz olduğu veri setidir. %99 negatif, %1 pozitif sınıf vb.

Precision-Recall Curve



Değişen threshold değerleri için PR nasıl değişir?

Sağ üste yaklaşan model daha iyidir.

Average Precision → PC Curve altında kalan alan

F1 score

Precision	Recall	F1
0.95	0.1	0.18
0.1	0.95	0.18
0.5	0.5	0.5
0.95	0.85	0.89

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Precision ve Recall arasında dengeli bir metrik

F1 harmonik ortalama mantığına dayanır. Aritmetik ortalamadan farkı değerler arasındaki büyük uçurumları daha fazla cezalandırır.

F1 score

Ground Truth	Prediction
0	0
0	1
1	1
1	0
2	2
2	2

Bir önceki slayt binary f1 score. Peki birden fazla sınıf varsa nasıl hesaplayacağız?

- **Her bir class için diğer classlar negatif kabul edilip hesaplanır**

F1 score

Ground Truth	Prediction
0	0
0	1
1	1
1	0
2	2
2	2

- **Macro F1** Her bir sınıf için F1 hesapla ortalama al

Class 0 F1 Score = 0.5

Class 1 F1 Score = 0.5 Macro F1 = ⅔ = 0.66

Class 2 F1 Score = 1.0

- **Micro F1** Her class için tp-fp-fn hesapla, topla, f1 hesapla

Class 0	Class 1	Class 2
tp: 1	tp: 1	tp: 2
fp: 1	fp: 1	fp: 0
fn: 1	fn: 1	fn: 0

TP = 4

FP = 2

FN = 2

Precision = 0.66

Recall = 0.66

Micro F1 = 0.66

F1 score

Ground Truth	Prediction
0	0
0	1
1	1
1	0
2	2
2	2

- **Weighted F1** Her sınıfı önceden belirlenmiş ağırlıklar ile çarp

Class 0 weight = 0.2

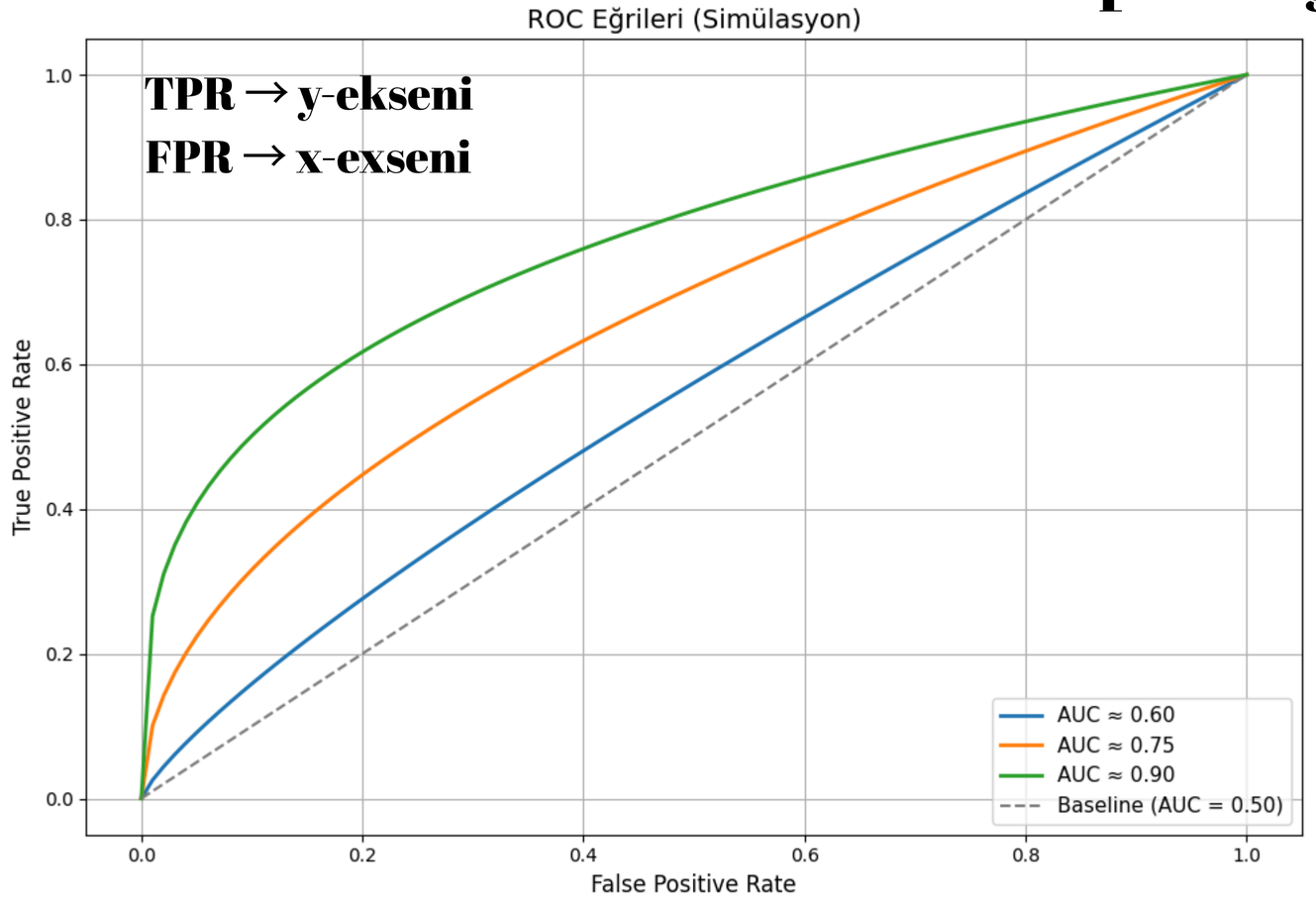
Class 1 weight = 0.3

Class 2 weight = 0.5

$$(0.2 \times 0.5) + (0.3 \times 0.5) + (0.5 \times 1.0) = 0.75$$

Weights veri dağılışına göre hesaplanabilir
weights her bir class için olabileceği gibi her bir örnek için de olabilir

ROC
Receiver operating characteristic



$$\text{True Positive Rate (TPR)} = \frac{\text{True Positive}}{\text{True Positive + False Negative}}$$

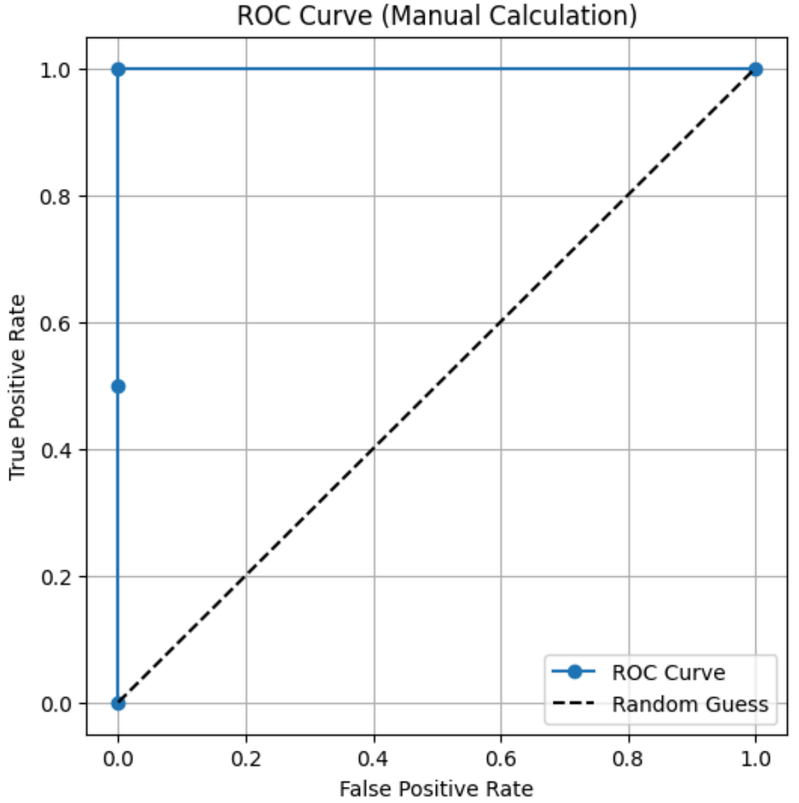
$$\text{False Positive Rate (FPR)} = \frac{\text{False Positive}}{\text{False Positive + True Negative}}$$

*sensitivity = Recall
*specificity = 1-FPR

- ROC eğrisi değişen threshold'lar için modelin performansını gösterir.
- Threshold sırası ile 0'dan 1' e kadar farklı değerlere eşitlenir, hesaplamalar yapılır ve grafik çizilir.
- Sadece olasılıksal çıktı veren modeller için kullanılır. (0-1 outputu veren modellerde eğri olmaz)
- 0-1 output veren modellerde (discrete classifier) eğri yerine xy ekseninde tek bir nokta olur.

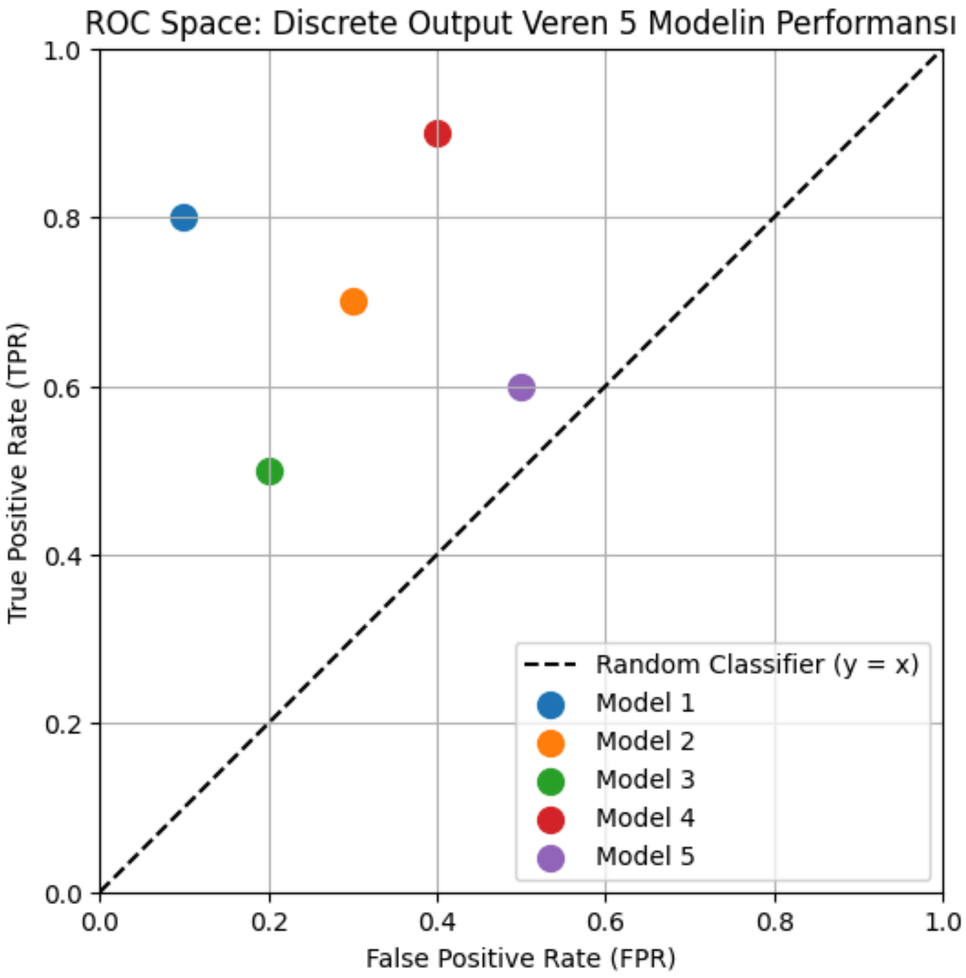
ROC
Receiver operating characteristic

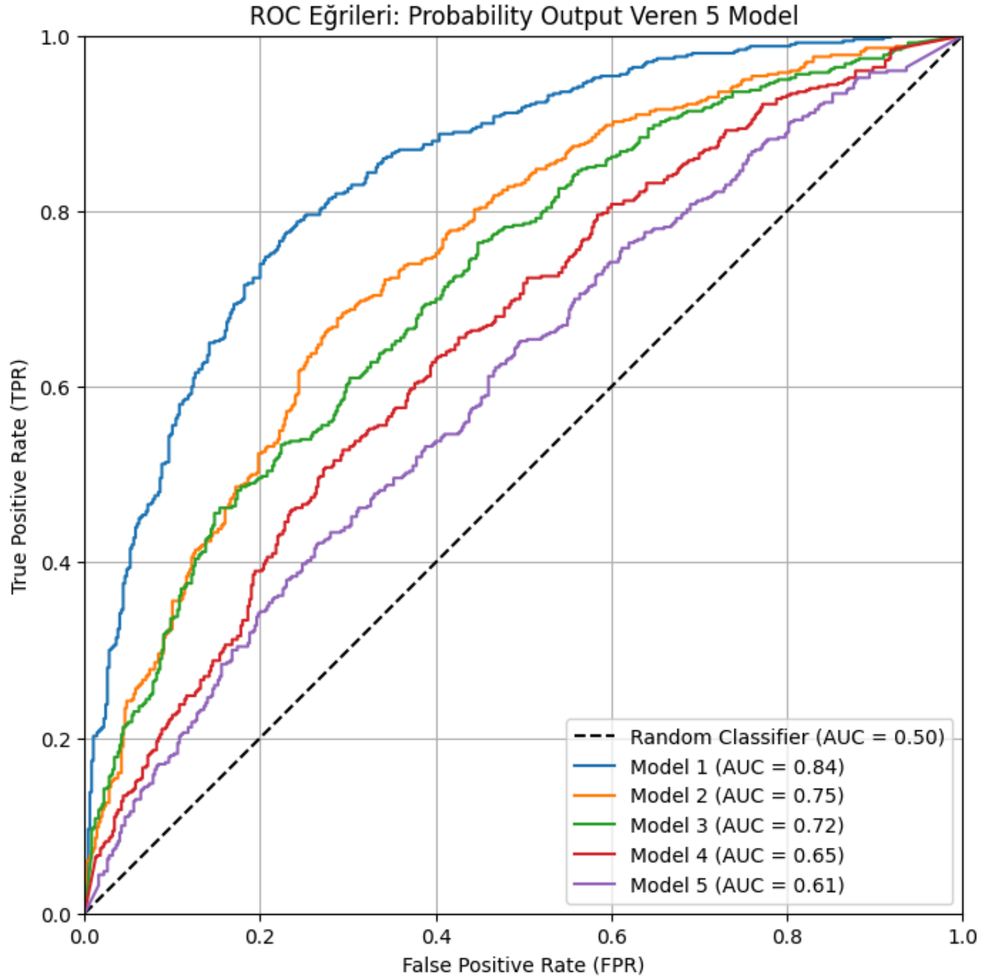
Ground Truth	Prediction	Eşik=0.1	Eşik=0.3	Eşik=0.5	Eşik=0.7
1	0.4	1	1	0	0
1	0.6	1	1	1	0
0	0.2	1	0	0	0
		TPR=1 FPR=1	TPR=1 FPR=0	TPR=0.5 FPR=0	TPR=0 FPR=0



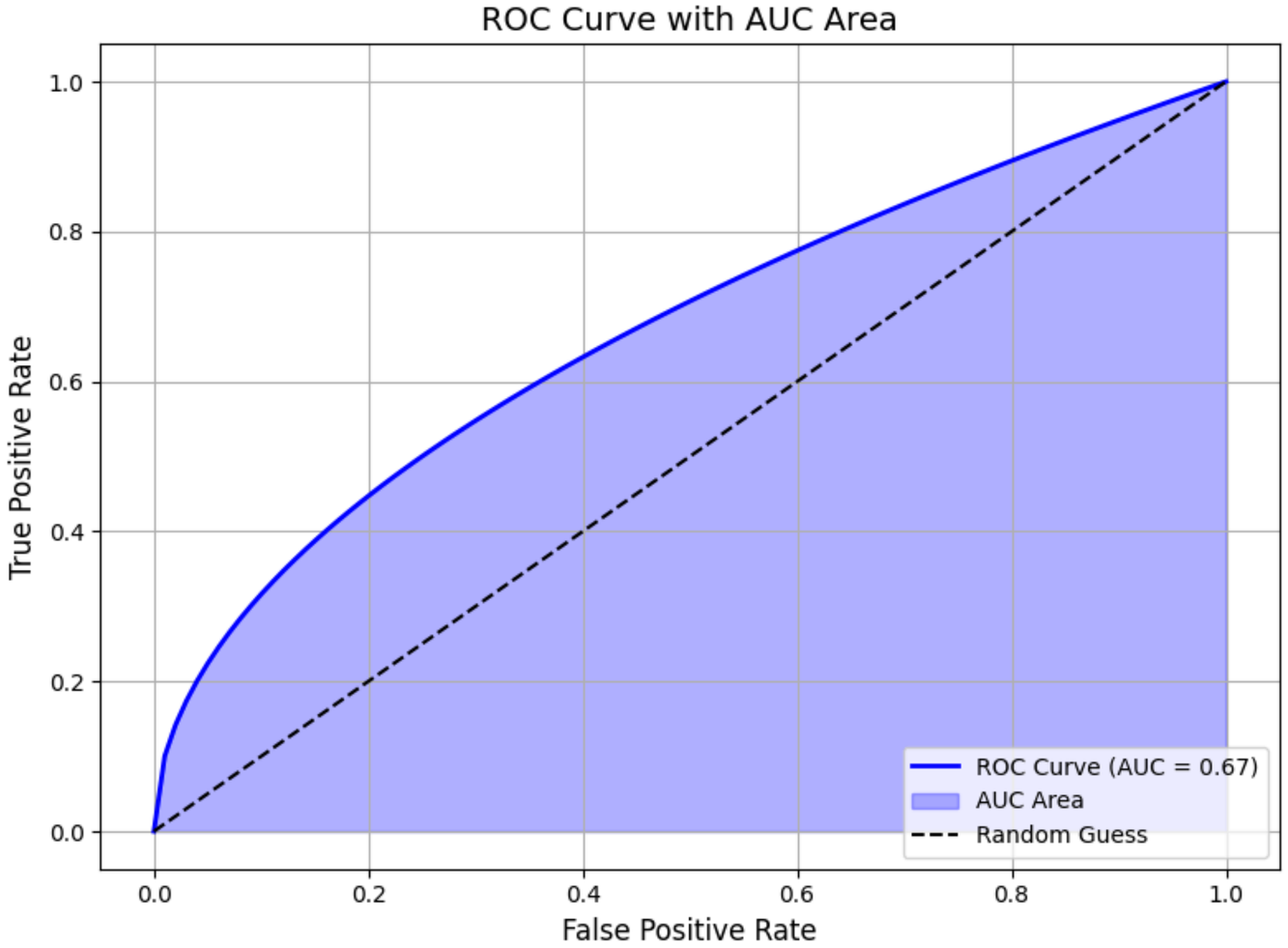
Sadece 3 veri olduğu için grafik mantıksal görünmeyebilir. Gerçek dünya verilerinde daha anlamlı grafikler elde edilir.

ROC
Receiver operating characteristic





AUC
Area under the curve



Roc eğrisinin altında kalan alandır
Tek bir sayısal değer ifade eder. Bu yüzden yorumlaması kolaydır.

Yüksek AUC → Daha iyi model

Farklı yöntemler ile hesaplanabilir

- **Trapezoidal Kuralı**
- **Mann-Whitney U Testi**
- **Simpson Kuralı ...**

Precision-Recall AUC → Precision-Recall eğrisinin altında kalan alandır. Birçok durumda Average precision ile aynı şeydir.

Train-Validation-Test

Train → modeli eğitmek için kullanılır
Validation → hyperparametreleri ve modelleri seçmek için kullanılır
Test → Seçilen sistemin nihai performansını değerlendirmek için kullanılır

- **10 bin adet veri olsun.**
- **6 bin adet train-2 bin adet validation-2 bin adet test**
- **3 farklı learning rate (0.1-0.01-0.001) değerinden hangisinin daha iyi sonuç vereceğini merak ediyoruz.**
- **3 farklı learning rate değeri olan 3 model oluştur. Her birini train verisi ile eğit.**
- **Her modeli validation verisi üzerinde değerlendir. En iyi performans veren modeli seç.**
- **Seçilen modeli test verisi üzerinde değerlendir ve nihai sonuca ulaş. Burada elde edilen değer proje veya çalışma sunumunda sayısal veri olarak kullanılabilir.**

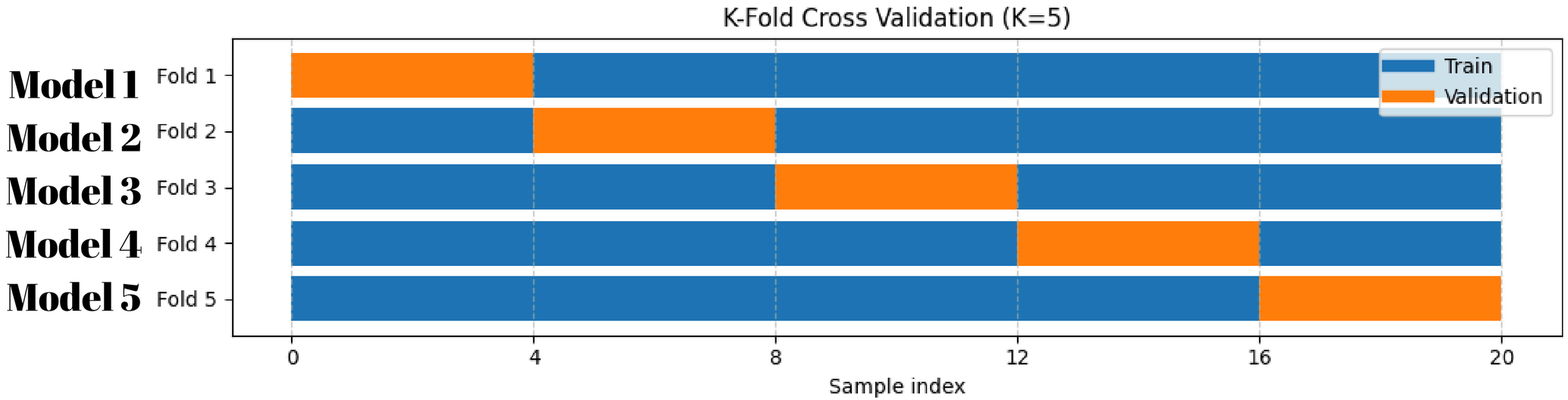
Train-Validation-Test

1. Validation ve test verisi model eğitimi için kullanılamaz. Bu verilerin eğitimde kullanılması modelin bu verileri iyi öğrenip gerçekçi olmayan sonuçlar vermesine sebep olur. (Data Leakage)
2. Validation verisi tarafsız sonuç bildiriminde (genellikle) kullanılmaz.
3. Test verisi olmadan da ML projesi geliştirilebilir. (Modeli eğit ve seç-Testleri gerçek hayatta yap)
4. Train-Val verileri modellerin performansı hakkında detaylı bilgi verir. Bu setler üzerinde uygun metrikler gözlenerek gelecekte daha iyi modeller eğitilebilir.
5. %60-20-20 ayrımı kesin değildir. Buradaki oranlar proje detayına göre şekillenebilir. (%99-1 bile olabilir)
6. 3 setin de veri setini gerçekçi yansıtması önemlidir (Dağılış-oran vb)
7. Basit Train-Validation ayrımının bazı eksikleri vardır. Validation setinin gerçeği yansıtmaması durumunda en iyi hyperparametreler seçilmeyebilir. Gerçeği yansıtsa bile tek bir tane validation setinin seçilmesi güvenilirliği azaltır.

Verilerin bu setlere uygun dağıtılması önemlidir. Sonraki konular bunla alakalı →

K-fold Cross Validation

Test verisi ayrıldıktan sonra (opsiyonel) kalan veri k parçaya bölünür. Her seferinde 1 parça validation verisi olarak kullanılır.



Bu yöntem Train-Validation ayrımına göre daha güvenilirdir. Burada toplamda 5 kere eğitim yapılıp 5 adet sonuç elde edilir. Bu yüzden validation performansı hakkında standart sapma-ortalama gibi yöntemler hesaplanabilir.

5 Adet model eğitilmesi demek eğitim süresinin yaklaşık 5'e katlanması demek. Bu yüzden büyük veri setlerinde ve işlem yükü yüksek modellerde k-fold kullanımı mantıksal olmayabilir.

K-fold Cross Validation

- k=5 olmak zorunda değil. Ancak genellikle 5 veya 10 seçilir. Bu seçim modelin büyüklüğüne, donanım yeterliliklerine göre şekillenebilir.
- 5 farklı model eğittikten sonra seçilen hyperparametreler ile son modeli elde etmek için
  - 5 modelin her biri kullanılıp çıktı alınabilir (y'). Her modelin verdiği outputun ortalaması alınabilir. (ya da ağırlıklı ortalaması alınabilir)
  - Train+Val verisinin tamamı Train olarak kullanılıp en iyi hyperparametreler ile tek bir model eğitilebilir .
  - 5 farklı modelden en ortalama olanı tek başına tahmin için kullanılabilir.

Stratified K-Fold

Fold'ların her birinde örnek oranının eşit olmasını sağlar. Geri kalan kısımlar k-fold ile aynıdır

	1. Fold	2. Fold	3. Fold	4. Fold	5. Fold	Toplam
Pozitif	1	1	1	1	1	5
Negatif	10	10	10	10	10	50

Veri setindeki class dağılımının Cross validation içerisinde korunması önemlidir. Model öğrendiği üzerinde test edilmelidir. Normal veri dağılımından farklı dağılım kullanmak optimistik ya da pesimistik skorlara sebep olabilir.

Repeated k-fold

K fold işleminin n kez tekrarlanması ile oluşur

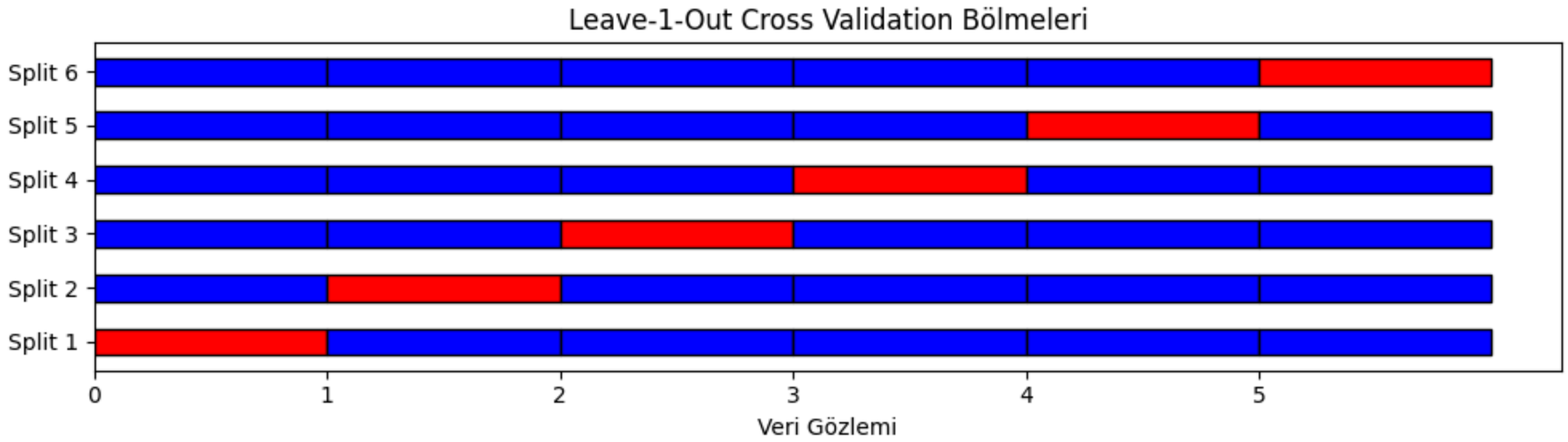
n=3 olarak seçilirse (k=5)

- Birinci turda veri 5 e ayrılır. k-fold işlemleri gerçekleştirilir. Sonuçlar elde edilir.
- İkinci turda veri farklı bir 5 gruba ayrılır. Aynı işlemler tekrar edilir.
- Üçüncü turda veri farklı bi 5 gruba ayrılır. Aynı işlemler tekrar edilir.

Sonuç olarak 3 turda da elde edilen sonuçların ortalaması alınır.

*Bu işlem k-fold'a göre daha güvenilir (robust) sonuç verir. Ama işlem yükünü n katına çıkarır.
*stratified Repeated k-fold ise ikisinin birleşimidir. n tur boyunca işlemler gerçekleştirilirken her turda veri dağılımının eşit olmasına dikkat edilir

Leave-p-out CV



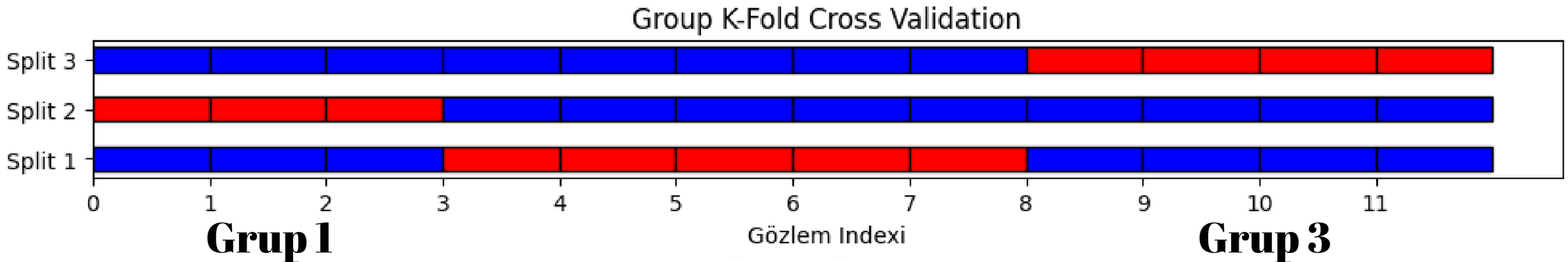
kırmızı bloklar 1 adet (tane) veri. Bir grup değil!!!

Her seferinde p tane veri CV için ayrılır. Bu işlemin büyük veri setlerinde uygulanması çok zordur.

*eğer p=1 ise özel olarak Leave-one-out CV denir.

1 milyon adet veri varsa → Leave-one-out için 1 milyon adet model eğitmek gerekir (mümkün değil)

Group K-Fold



Bir gruba ait veriler ya sadece train içerisinde ya da sadece val içerisinde bulunur.

Bir hastaya ait 100 adet farklı veri olduğunu varsayalım. Hasta belirli bir süredir bir hastalığa sahip (pozitif sınıf) ise farklı zamanda toplanmış 100 adet verinin her birinde benzer semptomlar görülecektir. Bu hastanın verileri ya sadece train ya da sadece val içerisinde olması gerekir. Diğer türlü train verisi içerisindeki değerler ile val verisi içerisindeki değerler kolayca eşlenerek gerçekçi olmayan ezberleme tahminlerin yapılmasına sebep olur.

***stratified group k-fold** ise grupların ayrılmasına ilkesini göz önünde bulundurarak verileri fold'larda eşit dağıtmaya çalışır.

Multilabel stratified K-Fold

Multilabel → Bir verinin birden fazla sınıfa ait olması (Bir beyin BT'sinin birden fazla hastalığı içermesi

Birden fazla label olduğu durumda her bir fold içerisinde her label eşit dağıtılmalıdır

Fold	1	2	3	4	5	Toplam
Class 0 + Class 1	1	1	0	1	0	3
Class 0	2	1	1	1	1	6
Class 1	1	1	1	1	1	5
Negatif	0	0	1	0	1	2

Class 0: Hipertansiyon
Class 1: Tip-2 diyabet

- Class 0 + Class 1 farklı bir class gibi ele alındı. Her fold içerisinde örnek bulunması sağlandı.
- Örnek sayısının yeterli olmadığı durumlarda bazı foldlarda temsili örnek olmadı (Negatif örnek sadece 2 fold içerisinde var.)
- Yeterli örnek olması durumunda her fold içerisinde her tipte örnek bulunabilir.

Confidence Interval

Tahmin ± Hata Payı

$$CI = \bar{X} \pm Z \cdot \frac{S}{\sqrt{n}}$$

$\bar{X}$: Örneklem ortalaması

Z : Z-tablosundan gelen kritik değer (örn: %95 için ≈1.96)

S : Örneklem standart sapması

n : Örneklem büyüklüğü

Fold	Accuracy
Fold 1	0.90
Fold 2	0.88
Fold 3	0.92
Fold 4	0.80
Fold 5	0.96

Bir tahminin ortalamadan ne kadar sapabileceğini ifade eder.
5-fold CV sonucunda ortaya çıkan 5 farklı sonuç bu yöntem ile ifade edilebilir.
5 fold'un ortalaması ile birlikte güven aralığını vermek akademiksel açıdan daha iyi bir tercihtir.

mean = 0.892

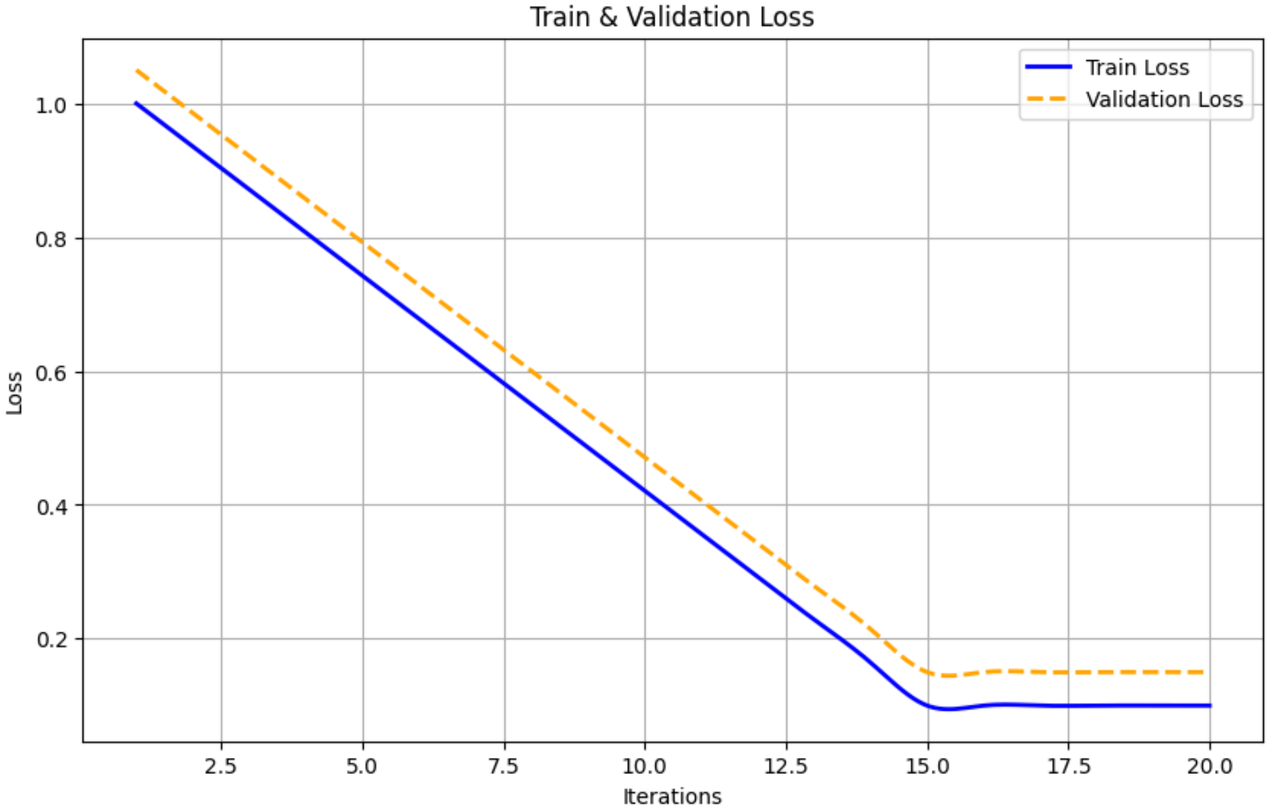
z = 1.96

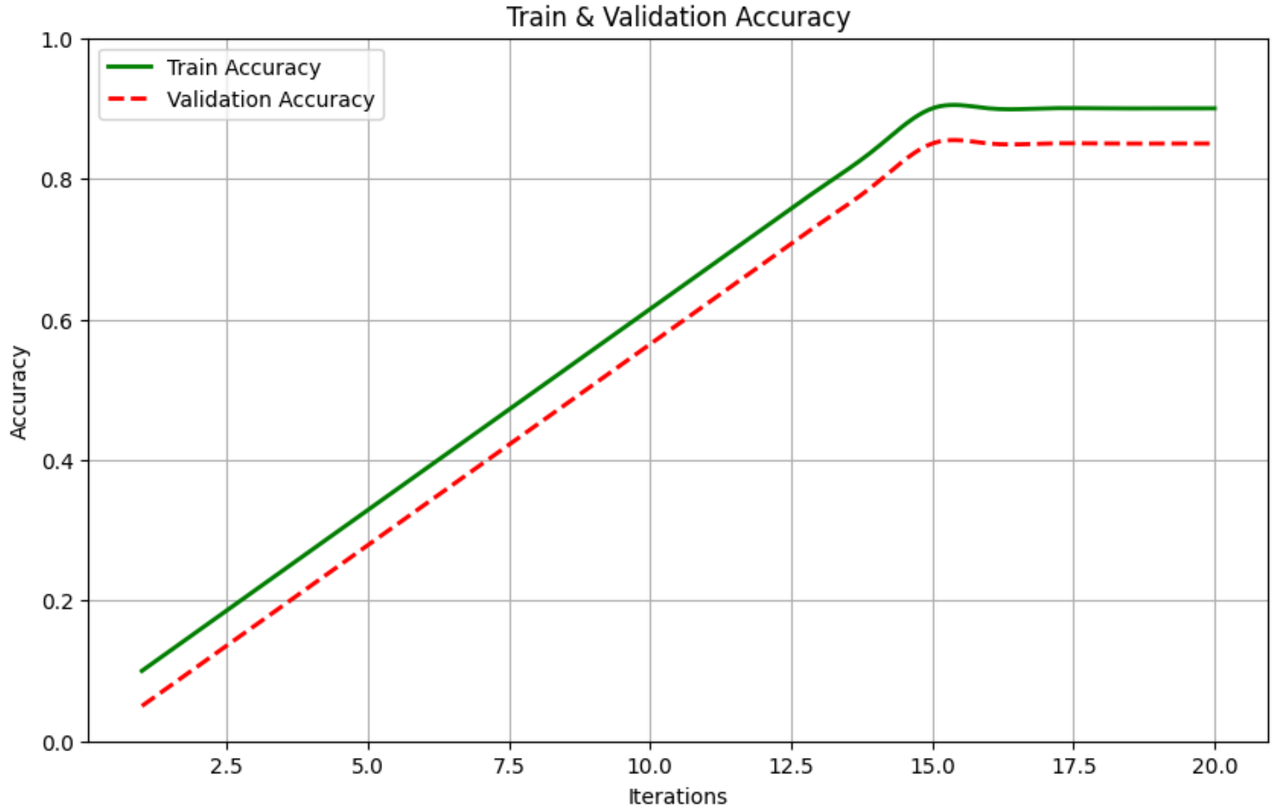
s= 0.059

n=5

0.892 +- 0.05

Train-Val Graphs





Her iteration için loss kaydet
Eğitim bittikten sonra grafiği çizdir

Her iteration için accuracy kaydet
Eğitim bittikten sonra grafiği çizdir

İki grafik de optimum bir eğitimin grafiğine benzemektedir. Eğer grafikler bu şekilde ise model iyi bir şekilde öğrenmiştir.

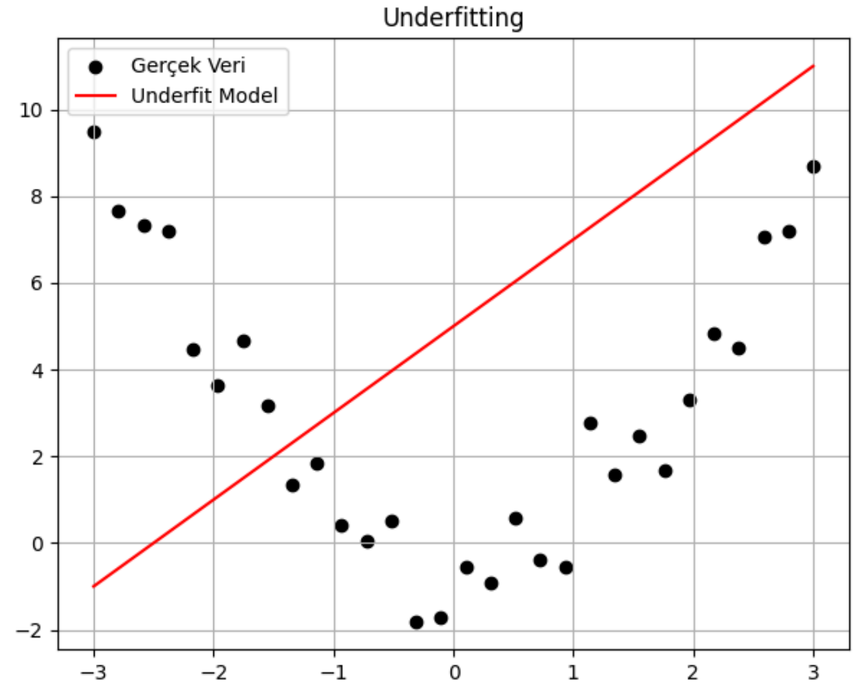
Bu grafikler eğitim anlaşılmasında çok önemlidir!!!

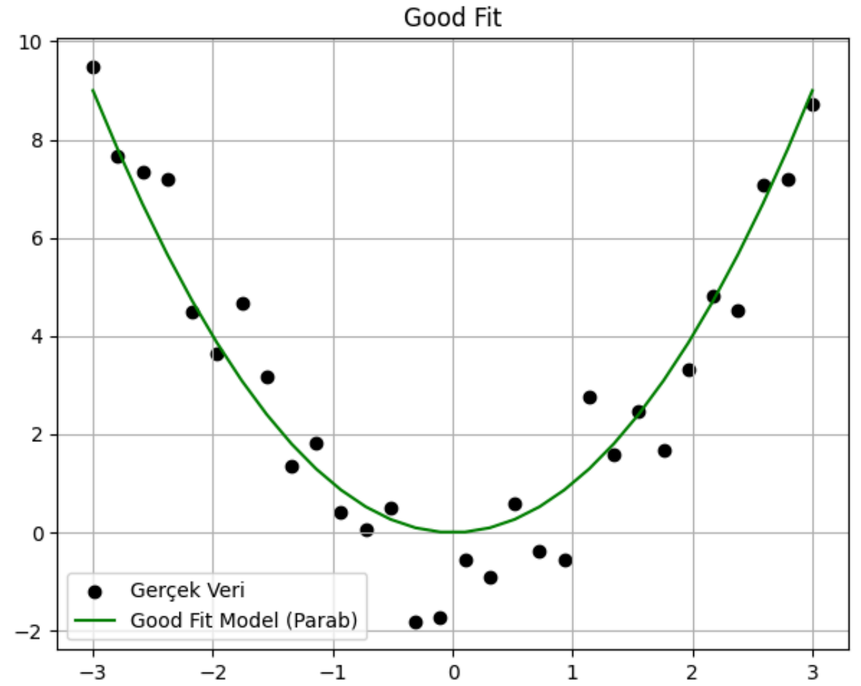
Genelleme

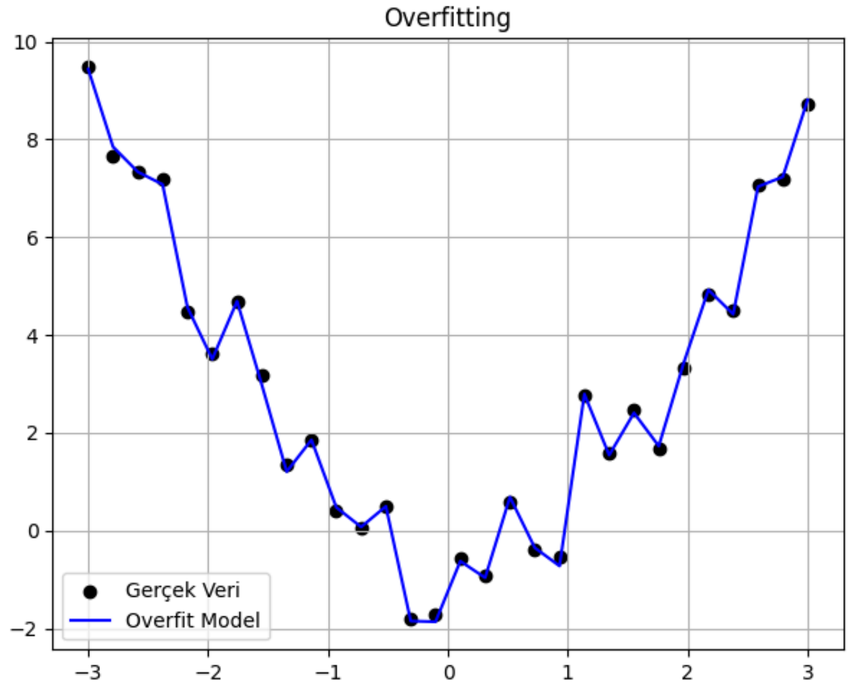
Model eğitimi ile elde edilmeye çalışılan kazanım modelin genelleme (Generalization) yeteneği kazanmasıdır. Train verisi üzerinde iyi öğrenip aynı performansı daha önce görmediği veri setleri (validation-test) üzerinde göstermesidir.

- İyi bir yapay zeka modeli (Right Fit) train verisi üzerinde genelleme yapabilmeyi öğrenir ve validation verisi üzerinde bunu uygulayabilir.
- Az öğrenmiş bir yapay zeka modeli (underfitting) yeterince öğrenemediği için hem train verisi üzerinde hem de validation verisi üzerinde yeterli performans gösteremez.
- Uydurarak öğrenmiş bir yapay zeka modeli (overfitting) genelleme yapmak yerine train verisi içerisindeki örnekleri ezberler. Bunun sonucunda train verisi içerisinde olmayan yeni bir veri ile karşılaştığında iyi performans gösteremez. Bu tür modellerde train değerleri yüksek iken val değerleri düşüktür.

Genelleme nasıl görünüyor







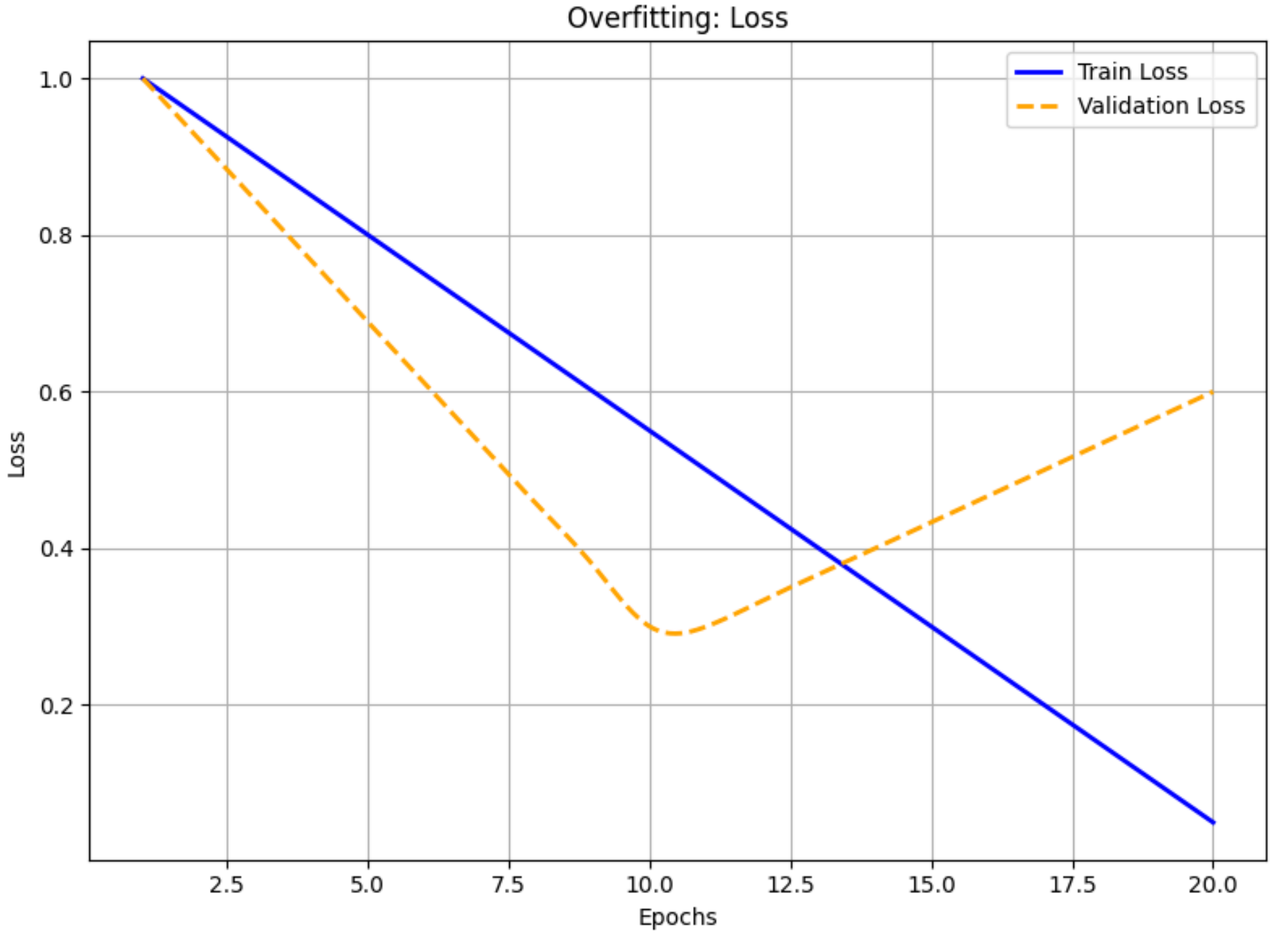
Yeterli öğrenememiş

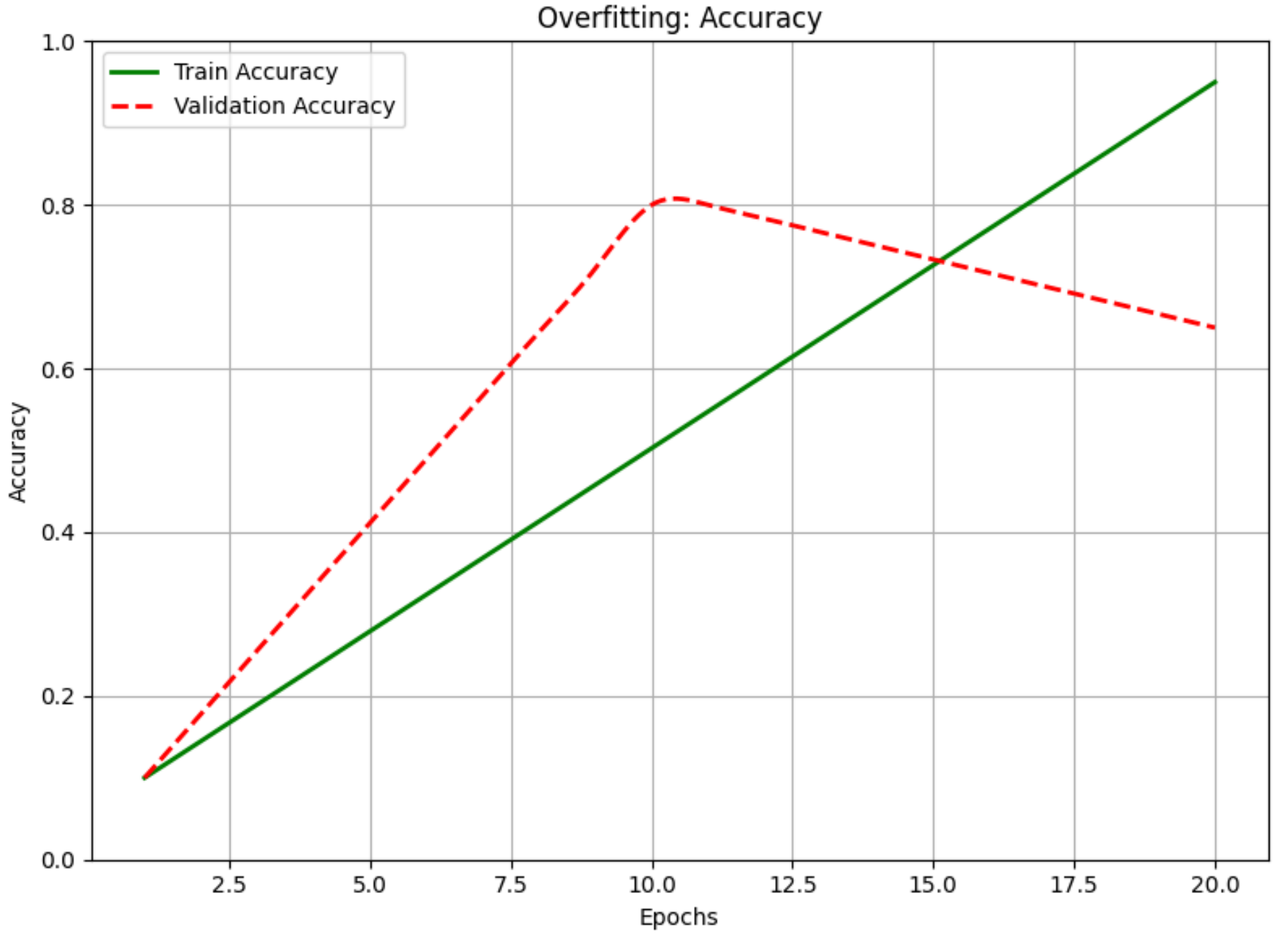
Güzel öğrenmiş

Uydurarak öğrenmiş

Görmediği yeni veride patlar

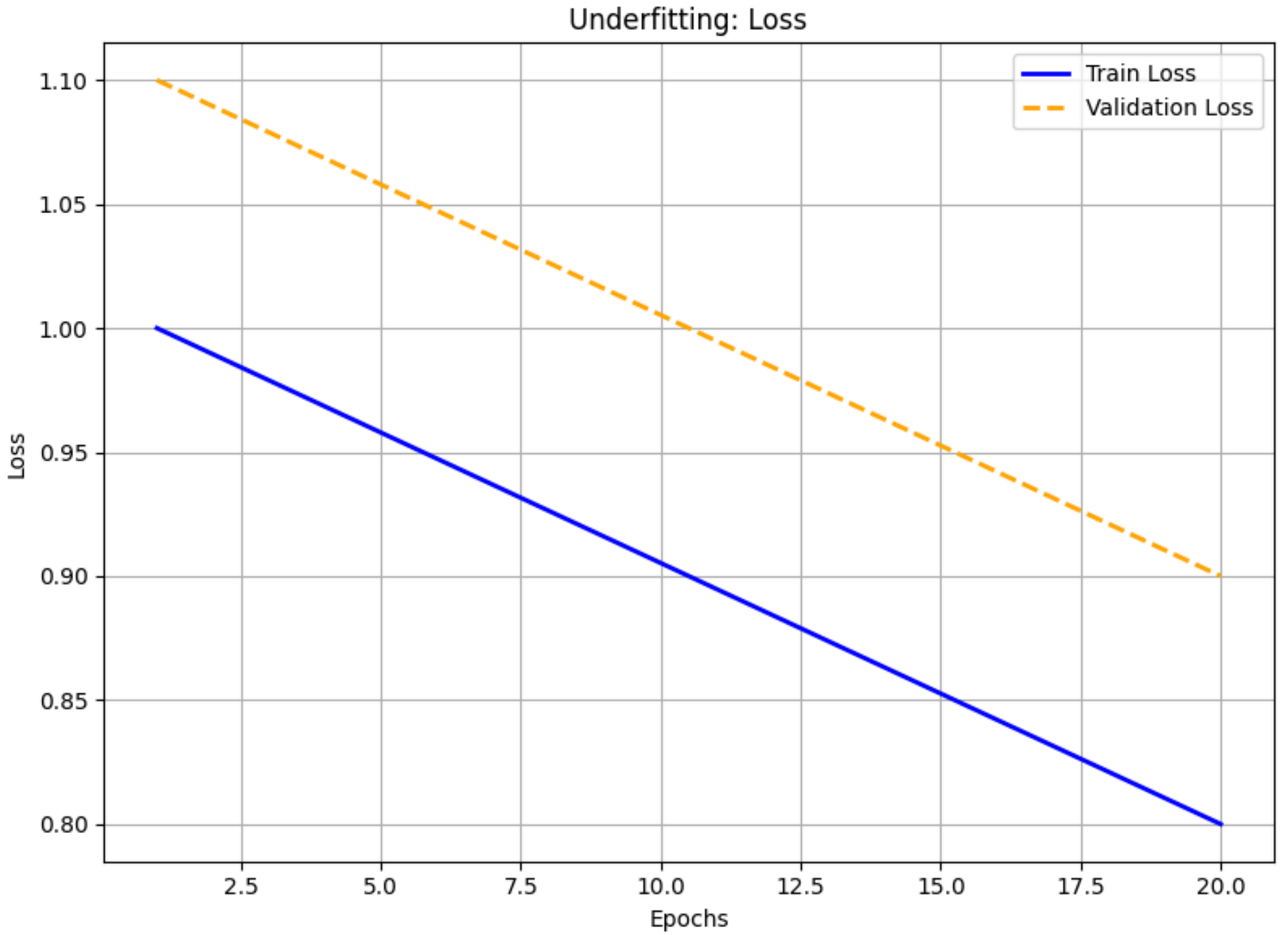
Overfitting

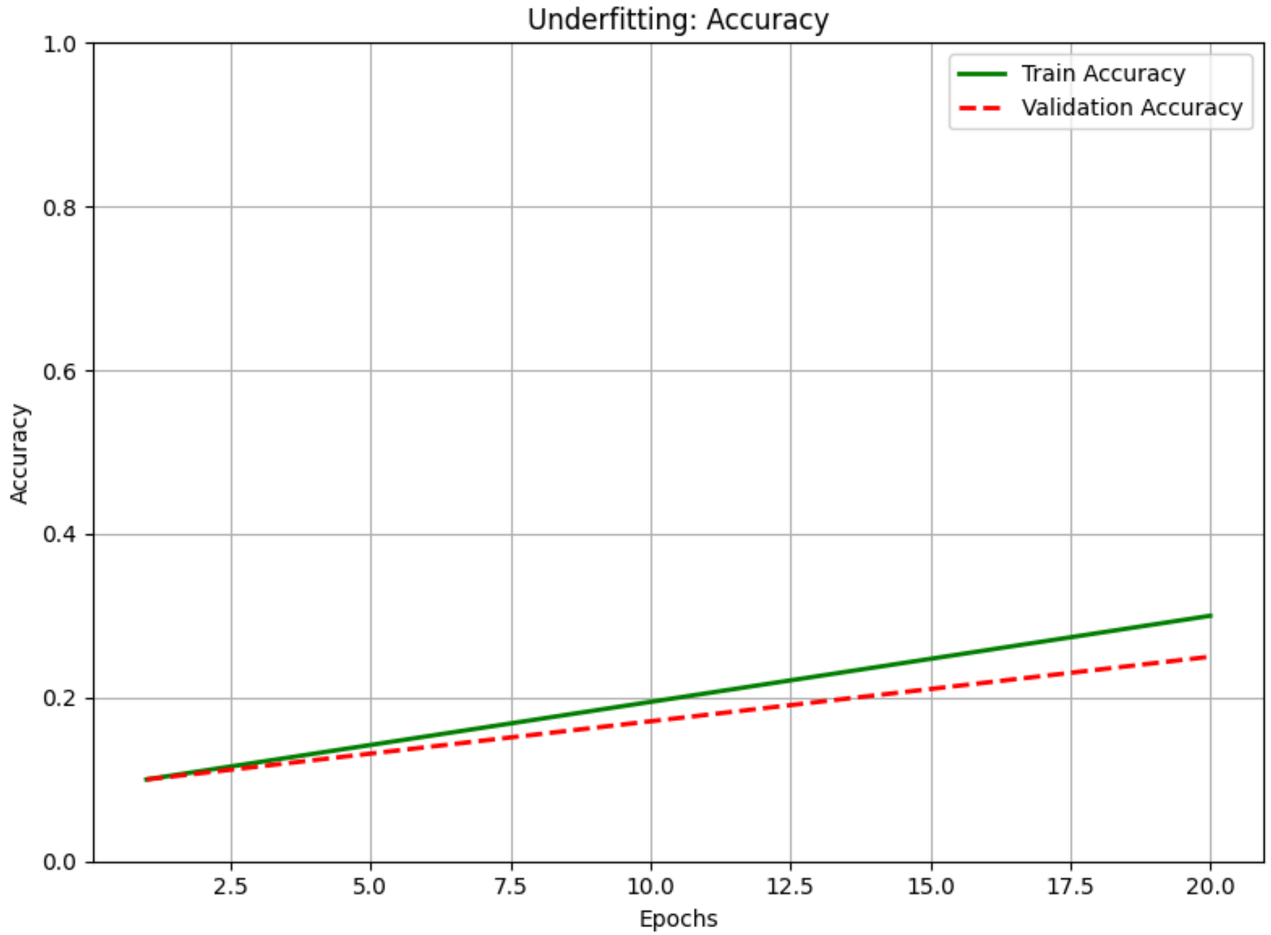




Eğitimin berirli bir aşamasından sonra validation loss artmaya ve validation accuracy düşmeye başlar.

Underfitting





Loss ve accuracy bir yere yakınsamamış, daha gidecek yolu var.
Eğitim devam ettirilirse daha fazla öğrenebilir.

Bazı underfitting durumlarında eğitim devam etse bile öğrenememe durumu olabilir. Bu da modelin sınırlarını gösterir.

Underfitting-Overfitting çözümleri

Underfitting	Overfitting
Model karmaşıklığını artırmak.	Model karmaşıklığını azaltmak
Regularization azaltmak	Regularization artırmak
Eğitim süresini uzatmak	Eğitim süresini azaltmak
Feature Engineering	Feature Selection
Dropout azaltma	Dropout artırma-Early stopping kullanma

Data Preprocess

Neural Network

Hyperparameter Search

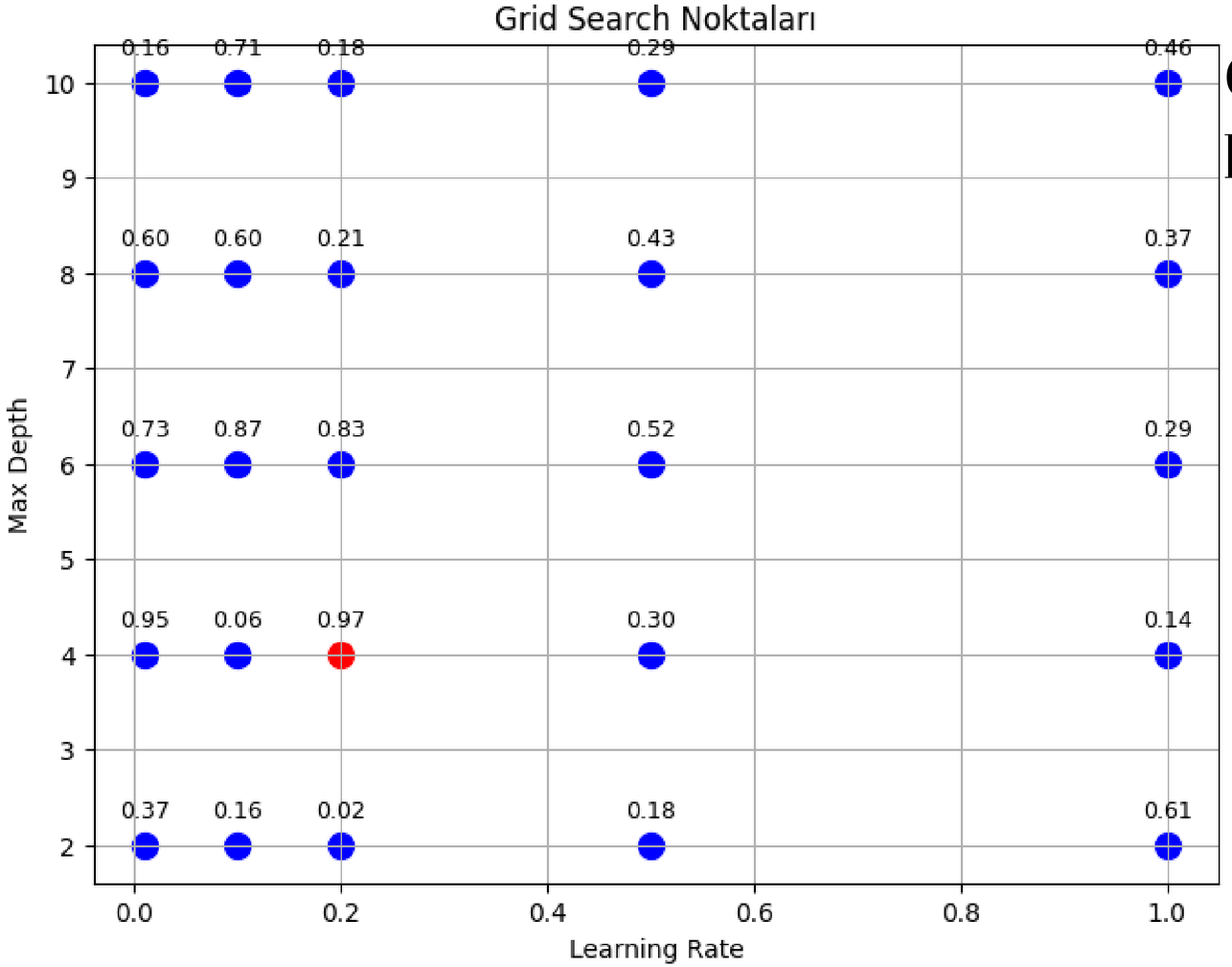
Parameter → Model tarafından eğitim sırasında güncellenerek öğrenilen sayılardır. (Ör: Weight-bias)

Hyperparameter → Modelin özelliklerini belirten parametrelerdir. Çoğu durumda model eğitimi sırasında değişmez. (Ör: Learning rate)

Model eğitimi sırasında learning rate kaç olarak seçilmeli: 0.1 - 0.01 - 0.001 ?
Hepsini deneyip en iyi sonuç vereni seçmek mantılı bir yöntem.

Hyperparameter search → Modelin önceden belirlenmiş hyperparametreleri için farklı değerleri deneyip en iyi sonucu seçme işlemi

Grid Search



Önceden belirtilmiş değerlerin her birini kombinasyon hesabı ile denemek

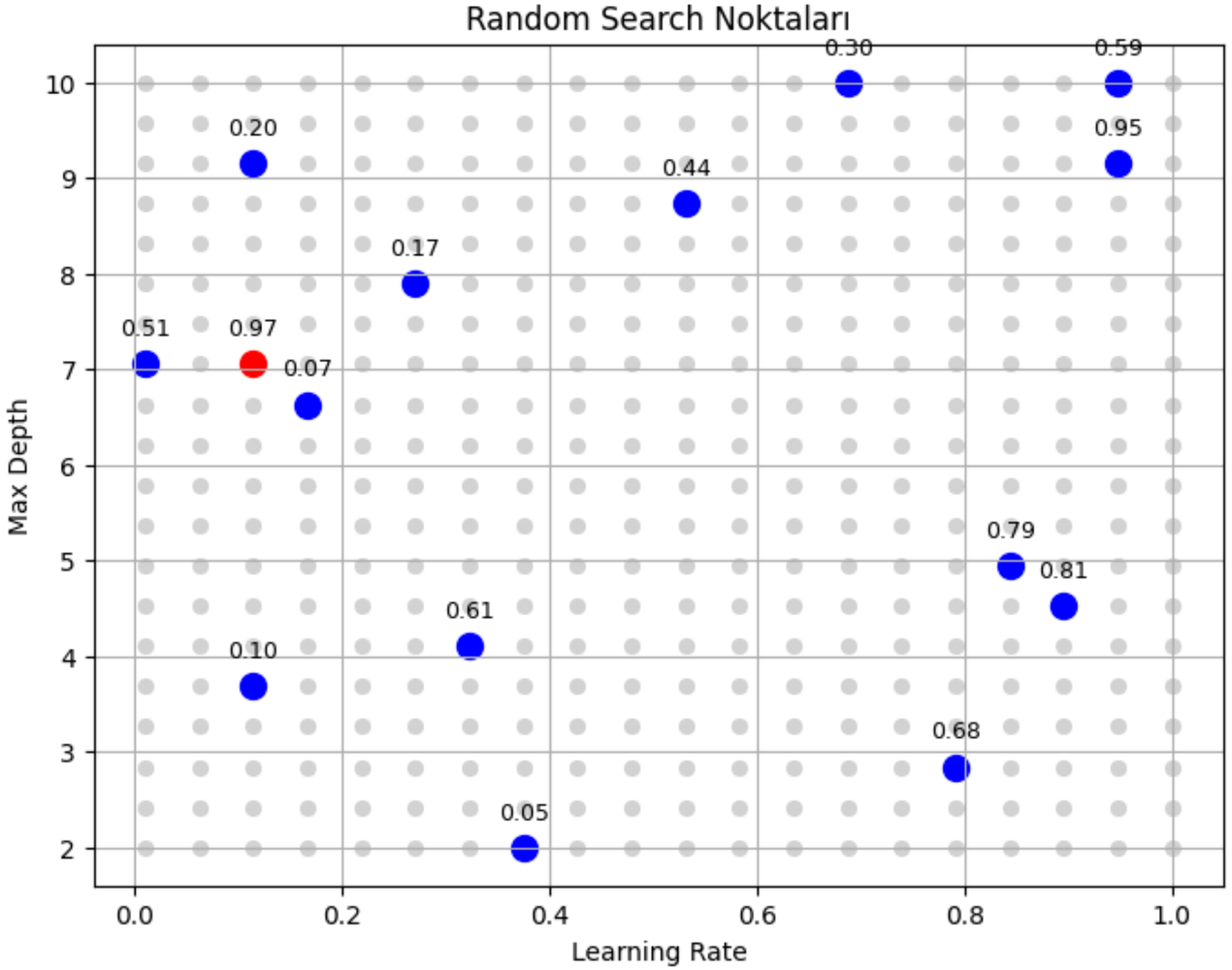
learning_rates = [0.01, 0.1, 0.2, 0.5, 1.0]
max_depths = [2, 4, 6, 8, 10]

Toplamda 5*5=25 adet model eğitildi. En iyi sonuç veren hyperparametreler seçildi.

Bu yöntem önceden belirlenmiş değerleri uyguladığı için ara değerler için işlem yapılmaz. Örnek olarak LR=0.05 değeri bu algoritmanın içerisinde denenmez.

Çok fazla değer denemek işlem sürecini çok fazla uzatabilir.

Random Search



Uzaydan random örnekleme yaparak en iyi noktayı bulmaya çalışır.

Geniş aralıklarda bilgi almak için kullnışlıdır. Aynı aralığı grid search ile taramak yıllar sürebilir.

Temel random search işleminde seçilen noktaların birbiri ile alakası yoktur. Noktaları birbiri ile ilişkilendirmek daha gelişmiş algoritmaların oluşmasını sağlar. İlerde →

Bayesian optimization

Random birkaç nokta seçerek başlar. Noktaların verdiği sonuca göre algoritmalar ile yeni seçeceği noktaları belirler.

- Optuna
- Scikit-Optimize (skopt)
- GPyOpt
- Hyperopt

Bayesian tabanlı algoritmanın nasıl çalıştığını açıklayan optuna'nın çıkış paper'ı
Optuna: A Next-generation Hyperparameter Optimization Framework

Overfitting to Cross-Validation

- Normalde CV seti overfitting olup omadığını ölçmek için kullanılır.
- Ancak Hyperparameter search algoritmalarının çok fazla nokta denemesi sonucu hiperparametre ayarları cross validation sonuçlarına göre defalarca optimize edilmiş olur. Bu durum CV setinin de ezberlenmesine ve CV sonuçlarının normalden daha iyi görünmesine yol açar.
- Eğer veri seti bozuk (corrupted) ve CV için ayrılan kısım küçükse bu ihtimal artar. Örnek olarak CV içerisinde hatalı örnek oranı %50 ise CV güvenilmez olur ve en iyi CV skoru gerçekten de en iyi olmayabilir. Aynı durumda CV setinin küçük olup tüm verileri yansıtmaması durumunda da olur.

Bozuk bir veri seti ve sabit küçük bir validation seti → Yüksek ihtimal
İyi bir veri seti ve repated k-fold → Düşük ihtimal

Preventing 'Overfitting' of Cross-Validation Data
By Andrew NG