**Ad:** Lütfi Yiğit

**Soyadı:** SAĞLAM

**Numara:** 190101090

**Ders:** Yapay Sinir Ağları (BSM430)

**Konu:** Müşteri kaybı tahmini

**Forward Propagation:** Input verisinin katmanlardan geçirilerek çıktının hesaplandığı süreçtir. Her bir katmandaki nöronlar, inputları ağırlıklarla çarpar ve bir aktivasyon fonksiyonu ile işlenir. Ağırlıklar başlangıçta rastgele atanır ve bu işlem boyunca değişmez.

**Back Propagation:** Ağın çıkışındaki hata ile başlayan bir süreçtir. Hata, çıkış katmanından başlayarak geriye doğru yayılır ve her bir katmandaki nöronların ağırlıklarına göre türevleri hesaplanır. Ağırlıkların güncellenmesi, gradient descent algoritmaları kullanılarak yapılır. Her bir ağırlığın güncellenmesi, öğrenme oranı ve hatanın gradienti ile belirlenir.

Yeni Ağırlık = Eski Ağıırlık − Öğ​renme Oranı × Hatanın Gradienti

**Learning Rate:** Learning rate, her güncellemede ağırlıkların ne kadar değişeceğini kontrol eden bir hiperparametredir. Learning rate’in optimal olması, modelin eğitim sürecinin başarılı olmasında önemli bir faktördür. Optimal öğrenme, yüksek öğrenme ve düşük öğrenmenin kendilerine göre avantajları ve dezavantajları vardır.

Optimal öğrenme;

Modelin minimum hataya daha etkili bir şekilde ulaşmasına yardımcı olabilirken, genellikle probleme özgüdür ve bu yüzden deneme yanılma yoluyla bulunması gerekir.

Yüksek öğrenme;

Eğitim süreci daha hızlı olabilir ama bu da salınımı sebebiyet verebilir. Model minimum hataya ulaşmadan önce aşırı salınım sebebiyle gidip gelebilir. Eğitim sürecinin patlamasına neden olabilir.

Düşük öğrenme;

Bu öğrenme tarzı ise daha istikrarlı bir eğitim süreci sağlar böylece daha düzenli ağırlık güncellemeleri anlamına gelir ama özellikle büyük veri setleri veya karmaşık modellerde çalışıldığında eğitim süreci uzayabilir ve modelin minimum hataya ulaşma şansı azalabilir.

**Loss Function:** Loss Function, modelin performansını değerlendiren, modelin tahminlerinin gerçek değerlerden ne kadar uzak olduğunu ölçen ve bu değeri minimize etmeye çalışan bir metriktir. Loss function değişikliğinde olası durumlar ise şu şekildedir;

Regresyon problemlerinde örneğin MSE kullanıldığında loss function artması durumunda modelin tahmin değerleri gerçek değerlere göre daha uzaklaşır, azalması durumunda ise gerçek değerlere daha yakınlaşır.

Sınıflandırma problemlerinde Binary Cross-Entropy Loss kullanıldığında artması durumunda sınıflandırma hataları artar, azalması durumunda ise sınıflandırma doğruluğu artar.

SVM problemlerinde Hinge Loss kullanıldığında artması durumunda modelin ayrım hatası artar, azalması durumunda modelin sınıfları daha iyi ayırabilme yeteneği artar.

**Epoch:** Model eğitilirken verilerin tamamı aynı anda eğitime katılmaz. Belli sayıda parçalar halinde eğitimde yer alırlar. İlk parça eğitilir, modelin başarımı test edilir, başarıma göre backpropagation ile ağırlıklar güncellenir. Daha sonra yeni eğitim kümesi ile model tekrar eğitilip ağırlıklar tekrar güncellenir. Bu işlem her bir eğitim adımında tekrarlanarak model için en uygun ağırlık değerleri hesaplanmaya çalışılır. Bu eğitim adımlarının her birine “epoch” denilmektedir.

Derin öğrenmede problemi çözecek en uygun ağırlık değerleri adım adım hesaplandığı için ilk epoch’larda başarım düşük olacak, epoch sayısı arttıkça başarım da artacaktır. Bununla birlikte belli bir adımdan sonra modelimizin öğrenme durumu oldukça azalacaktır.

Modelin eğitilmesi genelde uzun zaman almaktadır; eğitimi günler, aylar süren modeller vardır. Derin öğrenmede bu durum olağandır. Bu nedenle diğer hiper parametreler ile mümkün olduğunca eğitim süreci kısaltılmaya çalışılmaktadır.

Epoch sayısının büyüklüğü problem türüne göre de değişiklik göstermektedir. Örneğin örüntü öğrenilen RNN(Recurrent Neural Network)’lerde epoch sayısı diğer modellere göre daha büyük tutulmalıdır. Epoch sayısı arttıkça modelin başarımı gözle görülür oranda artmaktadır. Başarım belli bir epoch’dan sonra çok küçük birimlerde artış göstereceği için bu noktalarda eğitim sonlandırılabilir.

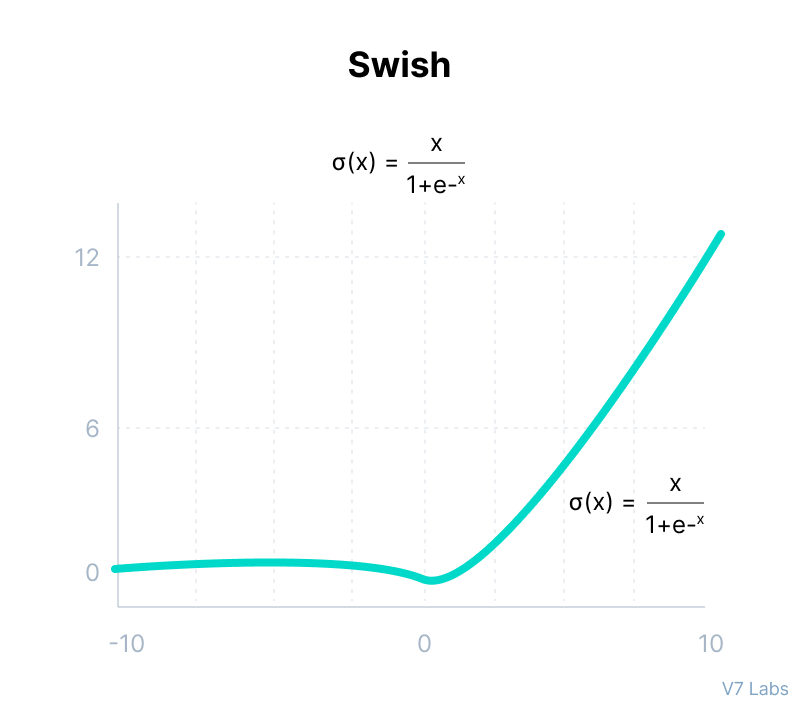
**Sigmoid Fonksiyonu:** Herhangi bir gerçel sayı aralığında bir değere eşleyen S-şekilli eğri oluşturan bir aktivasyon fonksiyonudur. Çıkış değerleri 0 ve 1 arasındadır ve genellikle sınıflandırma problemlerinde kullanılır. metin, diyagram, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Açıklama otomatik olarak oluşturuldu

**ReLU (Rectified Linear Unit) Fonksiyonu:** Giriş değeri pozitifse giriş değerini, negatifse sıfırı çıkaran bir aktivasyon fonksiyonudur. Özellikle derin sinir ağların kullanıldığında eğitim sürecini hızlandırabilir.

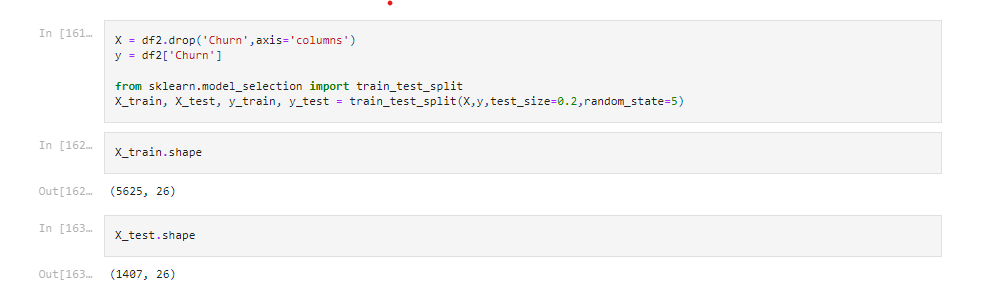
çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram, makbuz içeren bir resim

Açıklama otomatik olarak oluşturuldu

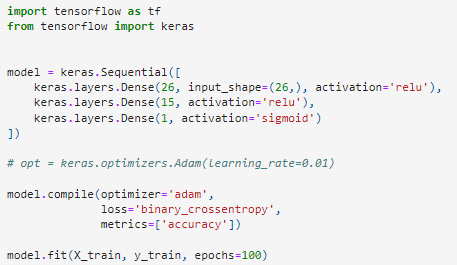
**Swish Fonksiyonu:** ReLU'nun doğrusal bölgesinin pozitif girişlerde daha yumuşak bir versiyonu ve sigmoid'in girişe bağlı olarak çıkışın sınırlarını belirleyen bir öğesi olan bir aktivasyon fonksiyonudur. Swish'in bazı durumlarda daha yumuşak gradyan geçişleri ve daha iyi genelleme yeteneği sağlayabileceği gözlemlenmiştir.

Matplotlib: Grafik oluşturmak ve görselleştirmek için kullanılan Python kütüphanesidir.

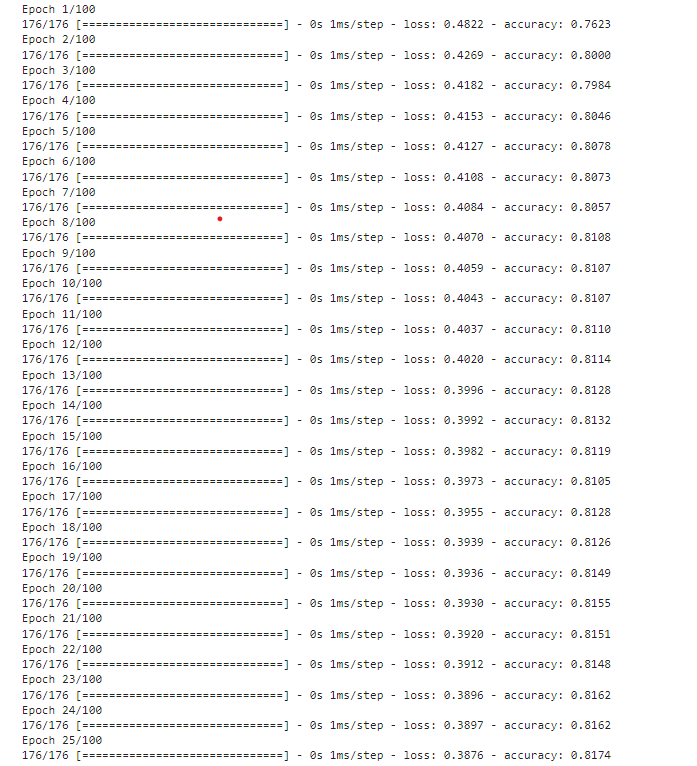
TensorFlow: Makine öğrenimi modelleri oluşturmak ve eğitmek için kullanılan açık kaynaklı derin öğrenme kütüphanesidir.

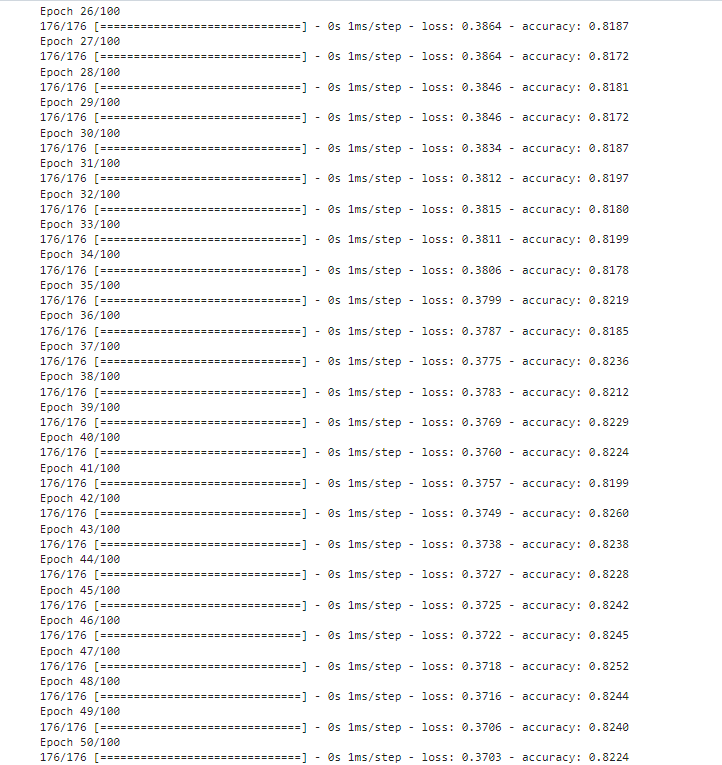


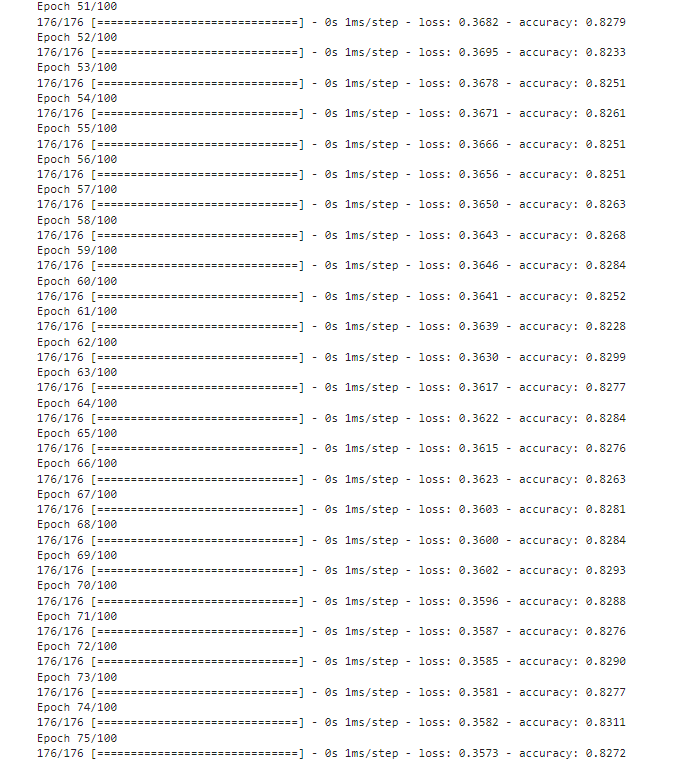
Buradaki kod satırlarında modelimizin verilerinin eğitim ve train şeklinde ayrımızı yapılıyor. test\_size=0.2 seçilerek verilerin %80’i train’e %20’si test işlemlerine ayrılıyor. Eğer test\_size=0.5 olsaydı %50’si train’e %50’si test işlemlerine ayrılırdı bu da daha büyük bir test seti, modelin performansını daha güvenilir bir şekilde değerlendirmenizi sağlar. Train seti boyutu düşer, ancak modelin daha genel bir performansını gözlemleme şansı artar. Dezavantaj olarak ise train seti boyutu küçüldüğü için modelin öğrenme süreci daha sınırlıdır. Eğer test\_size=0.1 olsaydı %90’ı train’e %10’u test işlemlerine ayrılmış olurdu bunun avantajı modelin performansını daha güvenilir bir şekilde değerlendirmenizi sağlar. Train seti boyutu düşer, ancak modelin daha genel bir performansını gözlemleme şansı artar. Dezavantajı ise train seti boyutu küçüldüğü için modelin öğrenme süreci daha sınırlıdır.

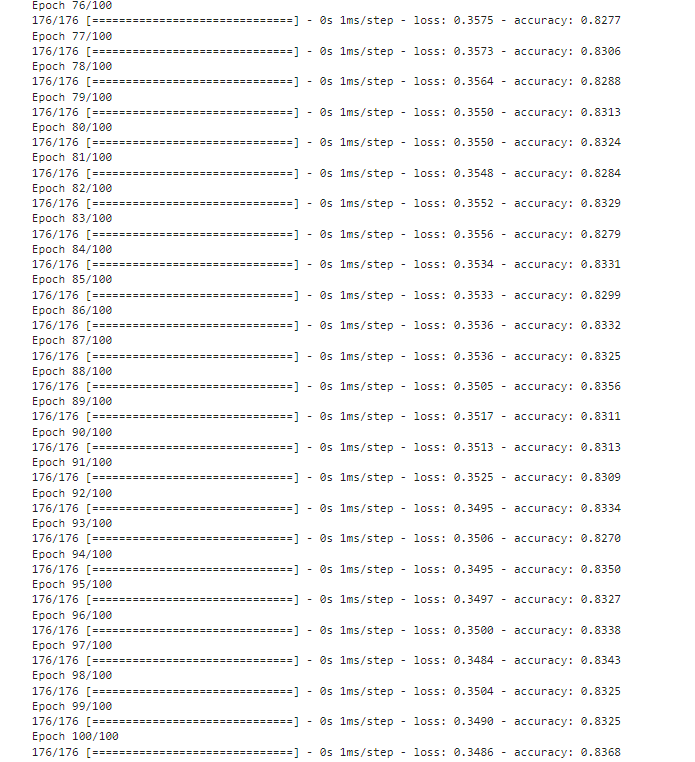


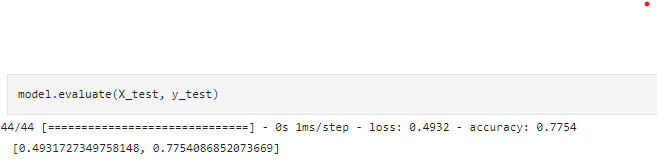
Bu kodda önce TensorFlow kütüphanesi ve Keras interface’i eklenir. Keras’ta modelin mimarisi tanımlanıp 3 katmanlı neural network kullanılıyor. İlk katman 26 input shape ile başlayıp sonrasında 15 nörona sahip aktivasyon fonksiyonu “ReLU” olan bir gizli katman ve son olarak aktivasyon fonksiyonu “Sigmoid” olan 1 nörona sahip çıkış katmanı bulunur. “model.compile” ile derlemesi yapılır. Optimizer olarak “adam” optimizasyon algoritması, loss function olarak binary crossentropy fonksiyonu kullanılır. “adam” optimizasyonun default learning\_rate değeri ise 0.01’dir. Eğer learning\_rate’i 0.1 olarak kullansaydık modelin eğitim sürecini hızlandırabilirdik ama çok yüksek learning\_rate olması aşırı dalgalanmaya veya modelin düzgün öğrenememesine neden olabilir. Eğer learning\_rate’i 0.001 olarak kullansaydık modelin daha istikrarlı bir şekilde öğrenmesine ve genelleme yeteneğini daha iyi korumasına yardımcı olabilir ama modelin eğitim sürecini yavaşlatabilir. “model.fit” ile modelin eğitimi belirtilen (100) epoch sayısı kadar eğitimini gerçekleştirir. Eğer burada epoch sayısını 50 olarak belirtseydik eğitim sürecimiz daha kısa olacaktı, modelimiz daha az sayıda eğitim geçireceği için aşırı öğrenmeye karşı daha dirençli olabilir ama yeterli öğrenme olmayabilir ve model belirli bir noktadan sonra daha fazla gelişme kaydedemeyebilir. Eğer epoch sayısını 150 olarak ayarlasaydık bu durumda da eğitim sürecimiz daha uzun sürecekti, model daha fazla eğitim verisi üzerinden öğrenme şansına sahip olacak ve daha fazla ayrıntıyı yakalayabilecektir ama aşırı öğrenme riskini arttırabilir ve modelin test seti üzerinde kötü performans göstermesine sebep olabilir.



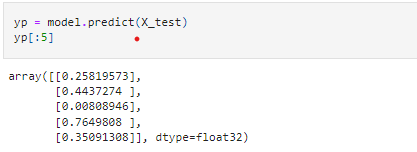




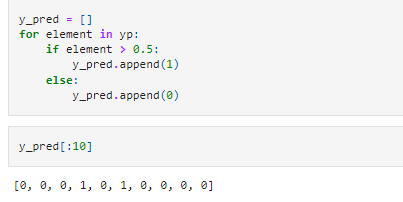




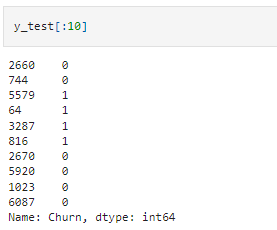
Bu kod ise modelin test seti üzerinde performansını ve doğruluğunu değerlendirir. Loss değeri 0.4932 gelmiştir. Düşük olması modelin test setinde daha iyi performans gösterdiğini gösterebilir. Yüksek olmasında ise tam tersi performans gösterdiğini gösterebilir. Accuracy değeri ise 0.7754 gelmiştir. Modelin test setinde doğru sınıflandırma oranını ifade eder. Bu değerin yüksek olması, modelin test setinde iyi performans gösterdiğini gösterir.



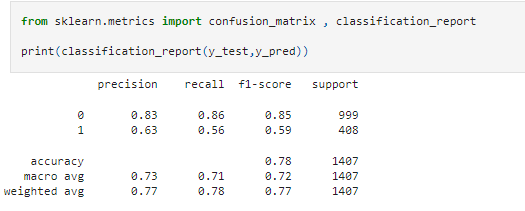
“model.predict(X\_test)” ile elde edilen modelin test seti üzerindeki tahmin olasılıklarını içeren bir yp oluşturur. “yp[:5]“ ifadesi ise bu dizinin ilk 5 öğesini gösterir. Çıktı ise modelin bir test örneği için yaptığı tahminin olasılık değerini içerir. Bu tahmin olasılıkları genellikle sigmoid aktivasyon fonksiyonu ile çıkış katmanında kullanıldığı için 0 ile 1 arasındadır. Eğer tahmin olasılığı 0.5'ten büyükse, genellikle bu örneği pozitif bir sınıfa tahmin etmiş oluruz, 0.5’ten küçük olursa negatif bir sınıfa tahmin etmiş oluruz.



Bu kod, önceki adımda “model.predict(X\_test)” ile elde edilen olasılık tahmin değerlerini (0 ile 1 arasında) bir eşik değeri olan 0.5 ile karşılaştırarak, ikili bir sınıflandırma problemi için tahmin sınıflarını belirler. Eğer bir örneğin tahmin olasılığı 0.5'ten büyükse, bu örnek "1" olarak sınıflandırılır; aksi takdirde "0" olarak sınıflandırılır.



Bu kod ise test setindeki gerçek değerleri gösterir.



Bu kod ise bize sınıflandırma raporunu vermektedir. Bizim modelimizin sınıflandırma raporu ise bu şekildedir.

Precision: Sınıf “0” için 0.83, Sınıf “1” için 0.63

Bir sınıfın doğru tahmin edildiği durumların toplam tahmin edildiği durumlara oranıdır.

Recall: Sınıf “0” için 0.86, Sınıf “1” için 0.56

Gerçekten sınıf "0" olan örneklerin yüzde 86'sının doğru bir şekilde tahmin edildiğini ifade eder. Sınıf "1" için, gerçekten sınıf "1" olan örneklerin yüzde 56'sının doğru bir şekilde tahmin edildiğini gösterir.

F1-Score: Sınıf “0” için 0.85, Sınıf “1” için 0.59

F1-score, precision ve recall değerlerinin harmonik ortalamasıdır. Dengeli bir performans ölçüsüdür. Sınıf "0" için, 0.85 değeri, precision ve recall değerlerinin dengeli bir şekilde yüksek olduğunu gösterir.

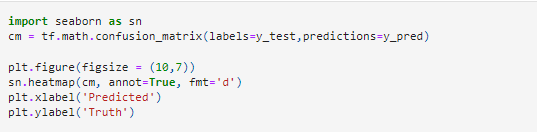
Support: Sınıf “0” için 999, Sınıf “1” için 408

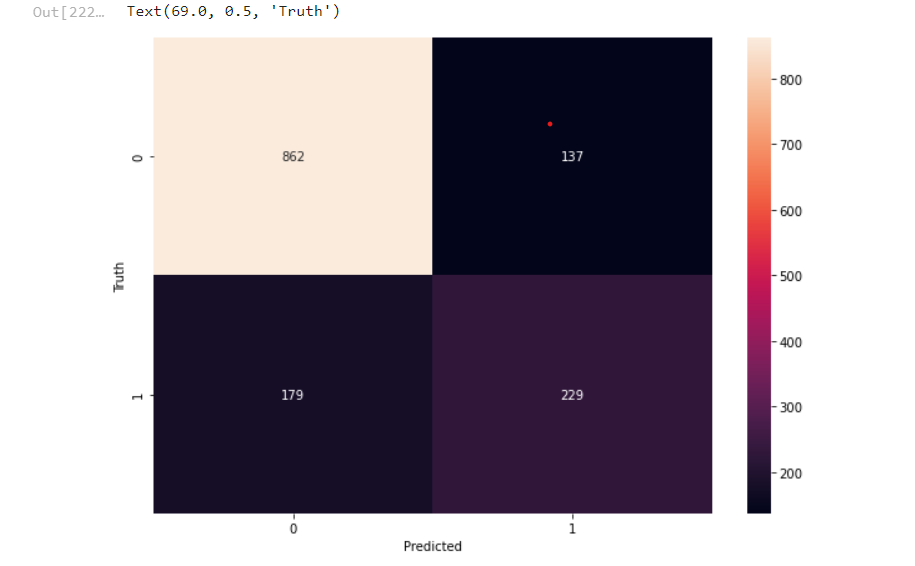
Her bir sınıftaki örnek sayısını ifade eder

Accuracy: Toplam doğru tahminlerin toplam örnek sayısına oranını ifade eder. Bu örnekte, 0.78 (78%) olarak gösterilmiştir.

Macro Average: Precision, recall ve f1-score değerlerinin sınıflar arasında eşit olarak ağırlıklı ortalama değeridir.

Weighted Average: Precision, recall ve f1-score değerlerinin sınıfların örnek sayılarına göre ağırlıklı ortalama değeridir.



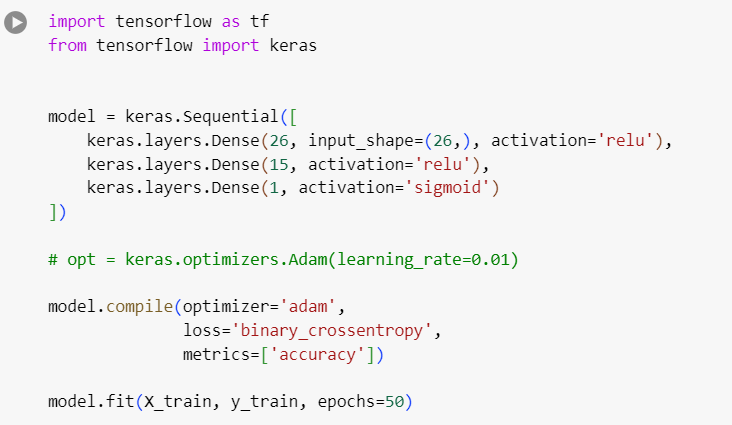


Truth 0, Predict 0 -> 862 veri. Modelin sınıf 0’ı doğru bir şekilde tahmin ettiği durumları gösterir.

Truth 1, Predict 0 -> 179 veri. Modelin sınıf 0 olarak yanlış tahmin ettiği durumları temsil eder. False Positive denir.

Truth 0 , Predict 1 -> 139 veri. Modelin sınıf 1 olarak yanlış tahmin ettiği durumları temsil eder. False Negative denir

Truth 1, Predict 1 -> 229 veri. Modelin sınıf 1’i doğru bir şekilde tahmin ettiği durumları gösterir.



metin, yazı tipi, makbuz, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Burada epoch değerinin 50 olarak güncellendiğinde loss değeri ve accuracy değerlerinin değişimi gözlenmiştir.

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

Classification report ise bu şekilde değişmektedir.

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

Truth predict tablosu ise bu şekilde değişmiştir.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Burada epoch değeri 150 olarak ayarlandığında ise aşağıdaki sonuçlar gözlemlenmiştir;

metin, ekran görüntüsü, yazı tipi, makbuz içeren bir resim

Açıklama otomatik olarak oluşturuldu

Loss değeri ve accuracy değeri bu şekilde değişmiştir.

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

Classification report bu şekilde değişmiştir.

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

Truth predict tablosu ise bu şekilde değişmiştir.

metin, yazı tipi, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Burada ise test size %50 olarak ayarlandığındaki çıktıdaki değişimleri göreceğiz;

metin, ekran görüntüsü, makbuz, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Loss değeri ve accuracy değerleri bu şekilde değişmiştir.

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

Classification report bu şekilde değişmiştir.

metin, ekran görüntüsü, yazılım, tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldu

Truth predict tablomuz ise bu şekilde değişmiştir.

KAYNAKÇA

<https://medium.com/operations-management-t%C3%BCrkiye/keras-loss-fonksiyonlar%C4%B1-2955e86a9e07>

<https://www.datacamp.com/tutorial/loss-function-in-machine-learning>

<https://medium.com/deep-learning-turkiye/derin-ogrenme-uygulamalarinda-en-sik-kullanilan-hiper-parametreler-ece8e9125c4>

<https://ayyucekizrak.medium.com/derin-%C3%B6%C4%9Frenme-i%C3%A7in-aktivasyon-fonksiyonlar%C4%B1n%C4%B1n-kar%C5%9F%C4%B1la%C5%9Ft%C4%B1r%C4%B1lmas%C4%B1-cee17fd1d9cd>

<https://medium.com/deeper-deep-learning-tr/ad%C4%B1m-ad%C4%B1m-forward-and-back-propagation-cf4cd18276ee>