***Ceren ÖNOL***

***Career HuBT – Veri Bilimi Eğitimi***

***Eğitim İsmi: Veri Bilimi***

***Eğitim Süresi: 60 Saat***

**Eğitim İçeriği**

1. **Makine Öğrenmesi**

**Makine Öğrenmesi Temelleri**

1. **Makine Öğrenmesi Nedir?**

Makine öğrenmesi, verilerden öğrenerek kararlar alabilen ve tahminlerde bulunabilen algoritmaların geliştirilmesini sağlayan bir yapay zeka dalıdır. Bu yöntem, sistemlerin deneyimle gelişmesini sağlar ve insanlar gibi öğrenme yetenekleri kazandırır.

**• Makine öğrenmesi vs Geleneksel Programlama**

Geleneksel programlamada, programcı kuralları ve mantığı belirleyerek sistemi kodlar; makine öğrenmesinde ise algoritmalar veri aracılığıyla bu kuralları kendileri keşfeder. Bu nedenle makine öğrenmesi, çok karmaşık ve kuralları önceden belirlenemeyen problemlerde daha etkilidir.

**• Hangi durumlarda kullanılabilir?**

Makine öğrenmesi, büyük miktarda veriden örüntüleri çıkarmak, tahminler yapmak ve karar almak gerektiğinde kullanılır. Özellikle tıp, finans, görüntü tanıma ve doğal dil işleme gibi alanlarda sıklıkla tercih edilir.

**2. Makine Öğrenmesi Çeşitleri**

**• Supervised, Unsupervised, Reinforcement Learning**

**Supervised (denetimli) öğrenmede, model eğitilirken doğru sonuçlar etiketlerle sağlanır ve model bu örneklerden öğrenir. Unsupervised (denetimsiz) öğrenmede etiketli veri kullanılmaz; model, veri içindeki gizli desenleri keşfetmeye çalışır, Reinforcement (pekiştirmeli) öğrenme ise ödül-ceza mekanizmasıyla çalışır ve model, doğru eylemleri keşfetmek için ortamla etkileşime girer.**

**• Semi-supervised ve Self-supervised Learning**

**Semi-supervised (yarı denetimli) öğrenme, az miktarda etiketli ve çok miktarda etiketlenmemiş veriyle çalışarak daha fazla bilgi çıkarımı sağlar. Self-supervised öğrenme ise veriden kendi etiketlerini oluşturarak kendini eğitir, bu yöntem özellikle veri etiketlemenin zor olduğu alanlarda kullanılır.**

**3. Veri İşleme**

**• Veri temizleme, özellik ölçeklendirme ve boyut azaltma teknikleri**

Veri temizleme, eksik veya yanlış verilerin ayıklanması ve düzeltilmesi sürecidir, modelin doğruluğunu artırır. Özellik ölçeklendirme, veriyi belli bir aralığa getirerek modelin daha hızlı ve doğru öğrenmesini sağlarken; boyut azaltma, gereksiz özellikleri kaldırarak modelin performansını iyileştirir.

**• Orantısız veri ile uğraşmak (imbalanced data set)**

Orantısız veri setlerinde bazı sınıflar diğerlerine göre daha az örneğe sahiptir, bu da modelin doğru tahmin yapmasını zorlaştırır. Bu sorunu çözmek için sınıf dengesini sağlamak üzere örneğin az olan sınıftan daha fazla veri eklemek veya özel algoritmalar kullanmak gerekir.

**4. Özellik Mühendisliği ve Seçimi**

**• Özellik çıkarımı teknikleri**

Özellik çıkarımı, ham verilerden modelin öğrenmesine katkı sağlayacak yeni bilgiler veya değişkenler oluşturma sürecidir. Bu işlem, verinin anlamlı yapılar haline getirilmesini sağlayarak model performansını artırır.

**• Özellik seçim yöntemleri**

Özellik seçimi, modelin gereksiz veya az önemli özellikleri ele alarak daha etkili bir şekilde öğrenmesini sağlamak için en önemli değişkenleri seçme sürecidir. Bu yöntem, modelin hızını ve doğruluğunu artırırken karmaşıklığı azaltır.

**5. Model Değerlendirmesi ve Seçimi**

**• Validasyon ve test seti seçmenin önemi**

Validasyon ve test setleri, modelin doğruluğunu ve genellenebilirliğini test etmek için veri setinden ayrılan özel parçalardır. Bu ayrım, modelin eğitim verisinde öğrendiklerini bağımsız veri üzerinde test ederek gerçek hayattaki performansını ölçmeyi sağlar.

**• sub-sample ile veri seçimini hızlandırmak**

Sub-sampling, büyük veri setlerinden daha küçük bir örnek seçerek modelin hızlı ve daha verimli bir şekilde eğitilmesini sağlar. Bu yöntem, model eğitimi için zaman ve kaynak tasarrufu sağlarken genel veri yapısını da yansıtmayı hedefler.

**• sub-sample seçim yöntemi**

Sub-sample seçiminde, ana veri kümesinden rastgele veya belirli kurallar çerçevesinde örnekler alınarak daha küçük bir veri seti oluşturulur. Bu yöntem, büyük veri setleri üzerinde çalışırken veri yükünü azaltmak için idealdir.

**• Cross-Validation**

Cross-validation, veriyi birçok parçaya bölerek her bir parçayı sırayla validasyon seti olarak kullanarak modelin performansını test etme yöntemidir. Bu teknik, modelin farklı veri bölümlerinde nasıl çalıştığını ölçerek daha güvenilir sonuçlar elde edilmesini sağlar.

**• Model karşılaştırma metrikleri**

Model karşılaştırma metrikleri, modellerin performansını kıyaslamak için kullanılan doğruluk, F1 skoru, hassasiyet gibi ölçütlerdir. Bu metrikler, hangi modelin daha iyi performans gösterdiğini anlamak için analiz yapılmasını sağlar.

**B. Denetimli Öğrenme Teknikleri**

**Sınıflandırma**

**6. Sınıflandırma Temelleri**

**• Sınıflandırma nedir?**

Sınıflandırma, veriyi belirli kategorilere veya sınıflara ayırmak için kullanılan bir denetimli öğrenme tekniğidir. Bu yöntemle, model verilen girdiyi tanıyarak belirli sınıflardan birine tahmin etmeye çalışır, örneğin "spam" veya "spam değil" gibi.

**• Sınıflandırma Metrikleri (Accuracy, Precision, Recall, F1-Score)**

 *Accuracy*: Modelin doğru tahmin ettiği toplam tahmin oranını gösterir, ancak dengesiz veri setlerinde yanıltıcı olabilir.

 *Precision*: Modelin belirli bir sınıf için yaptığı doğru pozitif tahminlerin tüm pozitif tahminlere oranıdır, yanlış alarmları azaltmada önemlidir.

 *Recall*: Gerçek pozitiflerin ne kadarının doğru tahmin edildiğini gösterir, özellikle kaçırılmaması gereken sınıflarda önem taşır.

 *F1-Score*: Precision ve recall'un harmonik ortalamasıdır, dengesiz veri setlerinde daha güvenilir bir performans ölçütü sunar.

**7. Linear Modeller**

**• Logistic Regression**

Logistic regression, sınıflandırma problemlerinde kullanılan ve çıktıyı iki sınıf arasında bir olasılık değeri olarak tahmin eden bir lineer modeldir. Veriyi ayrıştırmak için sigmoid fonksiyonunu kullanır ve genellikle ikili sınıflandırma problemlerinde tercih edilir.

**• Support Vector Machines (SVM)**

Support Vector Machines, veriyi sınıflandırmak için en uygun ayrım çizgisini (veya hiperdüzlemi) bulmaya çalışan güçlü bir sınıflandırma algoritmasıdır. Özellikle karmaşık veri setlerinde ve sınıfların birbirine yakın olduğu durumlarda iyi performans gösterir.

**8. Ağaç Bazlı Modeller**

**• Decision Trees**

Decision Trees, veriyi özelliklere göre dallara ayırarak kararlar veren ve her dalda bir sınıf tahmini yapan bir sınıflandırma modelidir. Kolay anlaşılabilir ve görselleştirilebilir olmakla birlikte, aşırı uyum (overfitting) yapma eğilimindedir.

**• Random Forests**

Random Forests, birden çok karar ağacından oluşan bir ansamble yöntemidir ve her ağaç bağımsız olarak karar verir, ardından sonuçlar birleştirilir. Bu yöntem, tek bir ağaç modeline göre daha güvenilir ve genellenebilir sonuçlar elde edilmesini sağlar.

**• Gradient Boosting Machines (e.g., XGBoost, LightGBM)**

Gradient Boosting Machines, zayıf öğrenicilerden (karar ağaçları gibi) oluşan bir ansamble modelidir ve her yeni ağaç, önceki ağaçların hatalarını düzeltmeye odaklanır. XGBoost ve LightGBM gibi versiyonları, modelin hızını ve doğruluğunu artıran optimizasyonlar ve düzenleme teknikleri sunar.

**9. Yapay Sinir Ağları**

**• NN yapısı**

Yapay Sinir Ağı (Artificial Neural Network - ANN), insan beynindeki sinir hücrelerinin işleyişine benzer bir şekilde çalışan, bir giriş katmanından, bir veya daha fazla gizli katmandan ve bir çıkış katmanından oluşan bir modeldir. Her katmanda bulunan nöronlar, ağırlıklı bağlantılar aracılığıyla birbirine bağlanarak bilgiyi işler ve sonuca ulaşır.

**10. İleri Sınıflandırma Teknikleri**

**• Ensemble Methods**

Ensemble Methods, birden fazla modelin bir arada kullanılarak daha güçlü ve doğru tahminler yapılmasını sağlayan tekniklerdir. Bu yöntemler, farklı modellerin hatalarını birbirini dengeleyerek daha genel ve güvenilir sonuçlar elde edilmesini sağlar (örneğin, Random Forest ve Gradient Boosting).

**• Imbalanced Datasets**

Imbalanced datasets, sınıfların dengesiz olduğu veri setleridir, yani bir sınıf diğerine göre çok daha fazla örneğe sahiptir. Bu durum, modelin azınlık sınıfı doğru tahmin etmede zorlanmasına yol açar, bu yüzden veri dengeleme teknikleri (örneğin, SMOTE) veya özel algoritmalar kullanılır.

**Regresyon**

Regresyon, sürekli bir hedef değişkeni tahmin etmek için kullanılan bir denetimli öğrenme tekniğidir. Girdi verileri ile hedef değişken arasındaki ilişkiyi modelleyerek, sayısal tahminler yapmayı amaçlar (örneğin, ev fiyatları tahmini).

**11. Regresyon Temelleri**

**• Regresyon nedir?**

**• Regresyon metrikleri (MSE, RMSE, MAE)**

 *MSE (Mean Squared Error)*: Modelin tahmin ettiği değerler ile gerçek değerler arasındaki farkların karelerinin ortalamasını alır; yüksek MSE, modelin hata yaptığına işaret eder.

 *RMSE (Root Mean Squared Error)*: MSE'nin karekökünü alarak, modelin hata büyüklüğünü orijinal birimlerde ifade eder, bu da yorumlamayı kolaylaştırır.

 *MAE (Mean Absolute Error)*: Gerçek değerler ile tahmin edilen değerler arasındaki mutlak farkların ortalamasını alır; modelin hata büyüklüğünü daha basit bir şekilde ölçer.

**12. Lineer Regression**

**• Simple ve Multiple Linear Regression**

 *Simple Linear Regression*: Bu model, tek bir bağımsız değişkenin hedef değişkenle olan doğrusal ilişkisinin incelendiği regresyon türüdür. Bir doğrunun denklemiyle model kurularak, bir veri setindeki iki değişken arasındaki ilişkiyi tahmin eder.

 *Multiple Linear Regression*: Bu model, birden fazla bağımsız değişkenin hedef değişkenle olan doğrusal ilişkisini inceleyen regresyon türüdür. Birden çok özellik kullanılarak, daha karmaşık ilişkilerin modellemesi yapılır ve tahminler daha doğru hale gelir.

**• Regularization teknikleri (Ridge, Lasso)**

 *Ridge Regression*: Ridge, modelin karmaşıklığını kontrol etmek için L2 ceza terimi ekler, böylece modelin aşırı uyum yapmasının önüne geçer. Bu teknik, tüm özelliklerin katsayılarını küçültür, ancak sıfır yapmaz.

 *Lasso Regression*: Lasso, L1 ceza terimi ekler ve bazı özelliklerin katsayılarını sıfırlayarak onları modelden çıkarır. Bu yöntem, daha az özellikle daha yalın ve anlamlı modeller oluşturulmasını sağlar.

**13. Non-Linear Regression**

**• Polynomial Regression**

Polynomial Regression, verinin doğrusal olmayan ilişkilerini modellemek için kullanılan bir regresyon türüdür. Bu model, bağımsız değişkenleri polinom terimleriyle genişleterek daha karmaşık eğriler oluşturur ve doğrusal olmayan ilişkiyi daha iyi yakalar.

**• Decision Trees ve Random Forest**

 *Decision Trees*: Karar ağaçları, veriyi özelliklere göre dallara ayırarak, her dalda bir tahmin yapar ve doğrusal olmayan ilişkileri modellemek için etkilidir. Ağaç yapıları, özellikle karmaşık verilerde iyi performans gösterir.

 *Random Forest*: Random Forest, birden çok karar ağacının birleşimidir ve her ağacın bağımsız olarak yaptığı tahminleri birleştirir. Bu yöntem, modelin doğruluğunu artırarak daha güvenilir ve genel sonuçlar elde edilmesini sağlar.

**• Gradient Boosting Machines (e.g., XGBoost, LightGBM)**

Gradient Boosting Machines, zayıf öğrenicilerden (karar ağaçları gibi) oluşan bir ansamble modelidir. Bu tür modellerde, her yeni model önceki modelin hatalarını düzeltmeye odaklanır; XGBoost ve LightGBM gibi popüler teknikler, bu süreci hızlandırarak yüksek doğruluk ve verimlilik sağlar.

**14. Support Vector Regression (SVR)**

Support Vector Regression, doğrusal olmayan veri setleri üzerinde regresyon yapmak için kullanılan bir yöntemdir. SVR, veriyi daha yüksek boyutlu bir uzaya haritalayarak, doğrusal olmayan ilişkileri modellemeye olanak tanır ve en iyi genelleme gücünü sağlamak için veriyi destek vektörleri etrafında optimize eder.

**• Kernel Tricks**

Kernel tricks, doğrusal olmayan ilişkileri modellemek için kullanılan bir tekniktir ve veriyi yüksek boyutlu bir uzaya dönüştürürken hesaplama maliyetlerini azaltır. Bu yöntem, çeşitli kernel fonksiyonları (örneğin, polinom, RBF) kullanarak veriyi daha karmaşık bir biçimde dönüştürür, böylece SVR'nin doğrusal olmayan veri üzerinde çalışmasını sağlar.

**15. Regresyon İçin Yapay Sinir Ağları**

Yapay Sinir Ağları, regresyon problemlerini çözmek için de kullanılabilir. Özellikle, karmaşık doğrusal olmayan ilişkileri modelleme yeteneği sayesinde, çok katmanlı sinir ağları (MLP - Multilayer Perceptrons) hedef değişkeni tahmin etmek için güçlü bir yöntem sunar.

**C. Denetimsiz Öğrenme Teknikleri**

**Modeli kullanarak veri ile alakalı ileri düzey analiz yapmak**

Denetimsiz öğrenme, etiketlenmemiş veri ile modelleme yapmayı sağlar. Bu teknikler, veri setindeki desenleri, yapıları ve gizli ilişkileri keşfetmek için kullanılır; örneğin, kümeleme (Clustering), boyut indirgeme (Dimensionality Reduction) gibi yöntemlerle veriden anlamlı içgörüler çıkarılabilir. Bu analizler, veriyi daha iyi anlamaya ve gelecekteki modeller için daha sağlam özellikler elde etmeye yardımcı olur.

**16. Kümeleme Algoritmaları**

**• K-Means, Hierarchical Clustering, DBSCAN**

 *K-Means*: K-Means, veriyi belirli sayıda kümeye ayırmak için kullanılan bir algoritmadır. Veriler, en yakın küme merkezine (centroid) atanarak, her küme için ortalama hesaplanır ve iteratif bir şekilde optimize edilir.

 *Hierarchical Clustering*: Bu yöntem, veriyi hiyerarşik bir yapıda kümelemeye olanak tanır. Aglo-meratif (alt kümeler birleştirilir) ve Divizif (kümelemeler ayrılır) olmak üzere iki ana yaklaşım vardır.

 *DBSCAN (Density-Based Spatial Clustering of Applications with Noise)*: DBSCAN, verinin yoğunluk bölgelerini bulmaya çalışan bir algoritmadır ve gürültüyü (outlier) ayırarak kümeler oluşturur. Bu yöntem, özellikle farklı yoğunluklardaki kümeleri algılamada etkilidir.

**• Kümeleme performansının ölçümü**

Kümeleme algoritmalarının performansı, genellikle dışsal (veri etiketlerine dayalı) veya içsel (veri içindeki yapıyı analiz ederek) ölçümlerle değerlendirilir. Dışsal ölçütler arasında Silhouette Score ve Adjusted Rand Index (ARI) yer alırken, içsel ölçütler arasında Davies-Bouldin Index ve Dunn Index kullanılır. Bu metrikler, kümeleme sonuçlarının doğruluğunu ve anlamlılığını belirlemede yardımcı olur.

**17. Boyut Azaltımı ve Manifold Learning**

**• Principal Component Analysis (PCA)**

Principal Component Analysis (PCA), verinin boyutunu azaltmak için kullanılan bir tekniktir. PCA, verideki en büyük varyansı açıklayan doğrusal bileşenleri (principal components) bulur ve verinin boyutunu bu bileşenler üzerinde projeksiyon yaparak azaltır, böylece daha düşük boyutlu bir veri seti elde edilir.

**• t-SNE, UMAP for Non-linear Dimensionality Reduction**

 *t-SNE (t-distributed Stochastic Neighbor Embedding)*, veriyi iki veya üç boyutlu uzaya indirgemek için kullanılan bir tekniktir ve özellikle verinin non-lineer yapısını koruyarak kümeleri görselleştirmeye yardımcı olur. t-SNE, özellikle yüksek boyutlu verilerin görsel analizi için uygundur.

 *UMAP (Uniform Manifold Approximation and Projection)*, t-SNE'ye benzer bir boyut indirgeme yöntemidir, ancak daha hızlıdır ve büyük veri setlerinde daha etkili çalışır. UMAP, verinin manifold yapısını koruyarak daha anlamlı ve ayrık düşük boyutlu temsiller oluşturur.

**18. Association Rule Mining**

**• Uygulamalar**

Association Rule Mining, veri madenciliği alanında, veriler arasındaki ilişkileri keşfetmek için kullanılan bir tekniktir. Bu yöntem, özellikle perakende sektöründe, müşterilerin birlikte satın aldığı ürünleri belirleyerek, çapraz satış stratejileri geliştirmeye yardımcı olur. Örneğin, "Ekmek alan müşteriler genellikle tereyağı da alır" gibi kurallar, pazarlama kampanyaları ve envanter yönetimi için kullanılabilir. Ayrıca, sağlık, bankacılık ve sosyal medya gibi farklı alanlarda da ilişkisel desenleri tespit etmek için kullanılır.

**19. Feature Importance**

Feature importance, bir modelin tahminlerinde hangi özelliklerin (özellikler) daha fazla etkili olduğunu belirlemek için kullanılan bir tekniktir. Bu metrik, her özelliğin modelin kararlarına katkısını ölçer ve daha anlamlı, etkili özelliklere odaklanılmasına yardımcı olur.

**20. Permutation Feature Importance**

Permutation feature importance, modelin tahminlerinin doğruluğunu değerlendirirken her bir özelliğin önemini belirler. Bu yöntem, bir özelliği rastgele karıştırarak modelin doğruluğundaki değişikliği ölçer; daha büyük bir doğruluk kaybı, o özelliğin daha önemli olduğunu gösterir.

**21. Partial Dependence**

Partial dependence, bir veya daha fazla özellik ile hedef değişken arasındaki ilişkiyi görselleştiren bir tekniktir. Bu yöntem, diğer tüm özellikler sabit tutularak, sadece bir özelliğin değeri değiştirildiğinde modelin tahmininin nasıl değiştiğini gösterir, bu da bir özelliğin hedef üzerinde nasıl bir etkisi olduğunu anlamaya yardımcı olur.

***D. Derin Öğrenme Temelleri***

**1. Derin Öğrenmeye Giriş**

**1. Derin Öğrenme Nedir?**

Derin öğrenme, yapay sinir ağlarını kullanarak verilerden öğrenmeyi sağlayan bir makine öğrenme alt alanıdır. Bu yöntem, özellikle büyük veri setleriyle çalışırken, modelin katmanlar aracılığıyla daha soyut ve derin temsiller öğrenmesine olanak tanır.

**• Makine öğrenmesi vs. derin öğrenme**

Makine öğrenmesi, verilerden belirli kalıpları öğrenerek tahminlerde bulunan bir tekniktir; genellikle daha az veriyle iyi sonuçlar verebilir. Derin öğrenme ise, çok katmanlı yapay sinir ağları kullanarak karmaşık veri yapılarından anlam çıkarır ve genellikle büyük veri setleri ve güçlü hesaplama gereksinimleri ile daha iyi sonuçlar elde edilir.

**• Derin öğrenmenin tarihçesi ve gelişimi**

Derin öğrenme, ilk olarak 1950'lerde yapay sinir ağlarının geliştirilmesiyle başladı, ancak hesaplama gücünün artması ve büyük veri setlerinin bulunmasıyla 2010'larda büyük bir ivme kazandı. Derin öğrenme, özellikle görüntü işleme, doğal dil işleme ve ses tanıma gibi alanlarda devrim yaratmıştır.

**2. Yapay Sinir Ağları Temelleri**

**• Aktivasyon ve türevlenebilir hesaplamalar kavramı**

Aktivasyon fonksiyonu, bir nöronun çıktısını belirlemek için kullanılan matematiksel bir fonksiyondur. Bu fonksiyon, ağın doğrusal olmayan ilişkileri öğrenmesini sağlar ve türevlenebilirliği, ağın geri yayılım (backpropagation) algoritması ile öğrenmeyi mümkün kılar, çünkü ağın hatalarını geriye doğru iletebilmek için türev hesaplanması gereklidir.

**• Nöron yapısı: girdi, ağırlıklar, bias, aktivasyon fonksiyonu**

Bir nöron, bir giriş (input), girişin ağırlıkları (weights), ve bir bias terimi ile çalışır. Girdi, modelin aldığı veriyi, ağırlıklar bu verinin model için ne kadar önemli olduğunu belirler, bias ise çıktının yerini ayarlayarak modelin doğruluğunu artırır. Aktivasyon fonksiyonu ise bu hesaplanan değerlerin, ağın çıktısını belirlemek için geçirdiği fonksiyondur (örneğin ReLU, Sigmoid, Tanh).

**3. Feed Forward Neural Network Yapısı**

**• Single layer perceptron**

Single Layer Perceptron (SLP), bir tek gizli katman ve bir çıkış katmanından oluşan basit bir yapay sinir ağıdır. Bu ağ, doğrusal karar sınırları oluşturur ve yalnızca doğrusal sınıflandırma problemlerini çözebilir; her bir girişin çıktıya olan etkisini ağırlıklarla belirler.

**• Multi layer perceptron (MLP)**

Multi Layer Perceptron (MLP), birden fazla gizli katman içeren ve her katmanda birden çok nöron bulunan bir yapıdır. MLP, doğrusal olmayan sınıflandırma ve regresyon problemlerini çözebilme yeteneğine sahiptir, çünkü her katman, daha karmaşık ve soyut özellikleri öğrenerek daha güçlü ve esnek modeller oluşturur. Bu yapı, ardışık katmanlar arasında aktivasyon fonksiyonları kullanarak veriyi işler ve öğrenir.

**2. Derin Öğrenme Bileşenleri**

**1. Aktivasyon Fonksiyonları**

**• Sigmoid, tanh, ReLU ve türevleri**

 **Sigmoid**: Sigmoid fonksiyonu, giriş değerini 0 ile 1 arasında bir değere dönüştüren S şeklinde bir fonksiyondur. Çıkışı, özellikle sınıflandırma problemlerinde olasılık tahminleri için uygundur, ancak gradyan kaybı sorununa (vanishing gradient) yol açabilir.

 **Tanh**: Tanh (hiperbolik tanjant) fonksiyonu, giriş değerini -1 ile 1 arasında dönüştürür. Sigmoid’e benzer, ancak sıfır etrafında daha güçlü gradyanlar sağlar, bu da öğrenmeyi daha hızlı hale getirebilir, fakat yine de vanishing gradient problemiyle karşılaşabilir.

 **ReLU**: ReLU (Rectified Linear Unit), negatif değerleri sıfıra indirirken pozitif değerleri olduğu gibi geçiren doğrusal olmayan bir aktivasyon fonksiyonudur. ReLU, hızlı öğrenme ve derin ağlarda iyi performans gösterir, ancak "ölü nöronlar" (dead neurons) sorununa yol açabilir.

 **ReLU Türevleri**: ReLU’nun türevi, giriş pozitifse 1, negatifse 0’dır. Bu basit türev hesaplama, geri yayılım sırasında hızlı ve verimli gradyan hesaplamalarına olanak sağlar.

**• Aktivasyon fonksiyonlarının karşılaştırılması**

 **Sigmoid ve Tanh**: Sigmoid, özellikle sınıflandırma problemlerinde kullanışlıdır, ancak sınırlı değer aralığı ve gradyan kaybı sorunları yaşanabilir. Tanh, sıfır etrafında daha güçlü gradyanlar sağlar, ancak yine de aynı vanishing gradient problemini barındırır.

 **ReLU**: ReLU, büyük derinlikli ağlarda daha hızlı öğrenmeye olanak tanır ve genellikle daha iyi performans gösterir. Ancak, negatif girişlerde gradyanların sıfırlanması nedeniyle ölü nöron problemi ortaya çıkabilir. Bununla birlikte, **Leaky ReLU** gibi türevleri bu sorunu kısmen çözer.

**2. Loss Fonksiyonları**

* **Mean Squared Error (MSE):** Modelin tahminlerinin gerçek değerlere olan farklarının karesinin ortalamasını alır, genellikle regresyon problemlerinde kullanılır.
* **Cross-Entropy Loss:** Sınıflandırma problemlerinde kullanılır ve modelin doğru sınıfı tahmin etme olasılığı ile gerçek etiket arasındaki farkı ölçer.
* **Hinge Loss:** SVM gibi modellerde kullanılan, doğru sınıfın diğer sınıflardan daha fazla uzak olmasını hedefleyen bir kayıp fonksiyonudur.

**3. Optimizasyon Algoritmaları**

* **Gradient Descent ve türevleri (SGD, Mini-batch GD):** Modelin parametrelerini, kayıp fonksiyonunun gradyanı doğrultusunda adım adım optimize eder. Stokastik gradient descent (SGD) ve mini-batch gradient descent gibi türevleri kullanılır.
* **Momentum, RMSprop, Adam:** Bu algoritmalar, gradient descent algoritmasının hızını ve doğruluğunu iyileştirir, özellikle daha büyük veri setleri ve karmaşık modeller için tercih edilir.

**4. Backpropagation Algoritması**

* **Zincir kuralı ve gradyan hesaplama:** Derin öğrenme modelinin her katmanındaki parametrelerin güncellenmesi için zincir kuralı kullanılarak gradyan hesaplanır.
* **Backpropagation adım adım uygulanması:** Modelin çıktı hatasından başlayarak her katmanda bu hata geriye doğru iletilir ve parametreler güncellenir.

**3. Model Eğitimi ve Değerlendirme**

**1. Veri Hazırlama**

* **Veri ön işleme teknikleri:** Veriyi temizlemek, eksik verileri işlemek ve özellik mühendisliği yapmak, modelin başarısını artırmak için gereklidir.
* **Normalizasyon ve standardizasyon:** Verinin belirli bir aralıkta ya da ortalama sıfır, standart sapma bir olacak şekilde düzenlenmesi, modelin daha hızlı ve verimli çalışmasına yardımcı olur.

**2. Eğitim Süreci**

* **Epoch vs Batch kavramları:** Epoch, tüm veri setinin model üzerinden bir kez geçmesi, batch ise verinin küçük parçalara bölünüp işlenmesidir.
* **Batch size seçimi ve etkileri:** Batch boyutu, modelin eğitim süresi ve genel başarısını etkileyen önemli bir parametredir.

**3. Overfitting ve Underfitting**

* **Bias-variance trade-off:** Modelin yeterince esnek olmaması (underfitting) ya da aşırı esnek olup veri gürültüsünü öğrenmesi (overfitting) arasında denge kurulmalıdır.
* **Regularizasyon teknikleri (L1, L2):** Aşırı öğrenmeyi engellemek için model parametrelerine ceza ekleyen tekniklerdir.

**4. Model Değerlendirme**

* **Train/Validation/Test set ayrımı:** Modelin genel performansını değerlendirmek için veri seti, eğitim, doğrulama ve test olmak üzere ayrılır.
* **Cross-validation teknikleri:** Modelin daha güvenilir bir şekilde değerlendirilmesi için k-katlamalı çapraz doğrulama kullanılır.

**4. İleri Derin Öğrenme Teknikleri**

**1. Batch Normalization**

* **Internal covariate shift problemi:** Eğitim sırasında ağırlıkların ve aktivasyonların dağılımındaki değişim, öğrenmeyi zorlaştırır. Batch normalization bu sorunu çözer.
* **Batch norm'un çalışma prensibi ve avantajları:** Aktivasyonların dağılımını sabitleyerek daha hızlı ve daha stabil bir eğitim sağlar.

**2. Weight Initialization**

* **Xavier/Glorot initialization:** Ağın ilk ağırlıklarını uygun bir şekilde başlatmak, ağın daha verimli öğrenmesini sağlar.
* **He initialization:** Derin ağlarda ReLU aktivasyon fonksiyonu kullanıldığında, ağırlıkların daha uygun şekilde başlatılması için tercih edilen bir tekniktir.

**3. Dropout**

* **Overfitting'i önlemede dropout'un rolü:** Eğitim sırasında bazı nöronları rastgele devre dışı bırakarak aşırı öğrenmeyi engeller.
* **Dropout uygulama stratejileri:** Genellikle %20-%50 oranında nöron bırakılır.

**4. Transfer Learning**

* **Pre-trained modeller ve fine-tuning:** Daha önce eğitilmiş modellerin, yeni bir görev için yeniden eğitilmesi, transfer learning olarak bilinir.
* **Transfer learning'in avantajları ve uygulama alanları:** Küçük veri setlerinde daha iyi sonuçlar elde etmek için kullanılır.

**5. Derin Öğrenme Mimarileri**

**1. Evrişimli Sinir Ağları (CNN)**

* **Evrişim işlemi ve özellik haritaları:** CNN'ler, görüntülerdeki önemli özellikleri öğrenmek için evrişim (convolution) işlemi kullanır.
* **Pooling katmanları:** Özelliklerin boyutunu küçültüp, modelin hesaplama yükünü azaltan işlemlerdir.
* **Klasik CNN mimarileri (LeNet, AlexNet, VGG):** Bu mimariler, evrişimli sinir ağlarının temellerini atmıştır.

**2. Sequence Modeling**

* **Tekrarlayan Sinir Ağları (RNN):** Özellikle zaman serisi verileri ve doğal dil işleme için uygun bir ağ yapısıdır.
* **LSTM ve GRU birimleri:** RNN'lerin sınırlamalarını aşmak için kullanılan, daha uzun süreli bağıntıları öğrenebilen gelişmiş yapılandırmalardır.
* **Bidirectional RNNs:** Zaman sırasıyla veriyi her iki yönden de işleyen bir ağ yapısıdır.

**3. Autoencoder'lar**

* **Encoder-decoder yapısı:** Autoencoder, veriyi sıkıştırıp daha sonra tekrar orijinal haline getirmeyi amaçlar.
* **Denoising autoencoder'lar:** Gürültülü veriyi temizlemek için kullanılan bir tür autoencoder modelidir.

**4. Generative Adversarial Networks (GAN)**

* **Generator ve discriminator kavramları:** GAN'lerde bir ağ, yeni veri örnekleri üretirken, diğeri bu örneklerin gerçek olup olmadığını kontrol eder.
* **GAN eğitim süreci ve zorlukları:** Eğitim sırasında generator ve discriminator birbirine karşı yarışarak gelişir.

**6. Derin Öğrenme Optimizasyonu**

**1. Hiperparametre Optimizasyonu**

* **Grid search ve random search:** Hiperparametrelerin en uygun değerlerini bulmak için kullanılan yöntemlerdir.
* **Bayesian optimizasyon:** Hiperparametre arama alanında daha verimli bir yaklaşım sunar.

**2. Model Compression**

* **Pruning teknikleri:** Ağaç yapılarındaki gereksiz dalların kesilmesiyle modelin boyutunun küçültülmesidir.
* **Quantization:** Modelin parametrelerinin daha düşük bitlerle temsil edilmesini sağlar.

**3. Gradient Clipping**

* **Exploding gradient problemi:** Modelin eğitiminde, gradyanların çok büyük olması sorununu çözmek için kullanılan bir tekniktir.
* **Clipping teknikleri ve uygulamaları:** Gradyanları bir eşik değeriyle sınırlayarak modelin stabilitesini artırır.

**4. Learning Rate Scheduling**

* **Step decay, exponential decay:** Öğrenme oranının eğitim süreci boyunca azalması, daha iyi genelleme yapmayı sağlar.
* **Cyclical learning rates:** Öğrenme oranının belirli aralıklarla artıp azaldığı bir stratejidir.

**7. Derin Öğrenme Frameworks**

**1. PyTorch**

* **PyTorch tensors ve autograd:** Tensors, veriyi işlemek için temel yapı taşlarıdır; autograd ise otomatik türev hesaplamasını sağlar.
* **PyTorch ile model tanımlama ve eğitim:** PyTorch, dinamik hesaplama grafiği kullanarak model geliştirme ve eğitim sürecini kolaylaştırır.

**2. TensorFlow ve Keras**

* **TensorFlow temelleri:** Derin öğrenme modellerini oluşturmak için geniş bir ekosisteme sahip bir framework'tür.
* **Keras API ile model oluşturma:** Keras, TensorFlow'un üst katman bir API'sidir ve hızlı prototip oluşturmayı sağlar.

**3. Framework Karşılaştırması**

* **TensorFlow vs PyTorch:** TensorFlow daha çok üretim ortamlarında kullanılırken, PyTorch araştırma ve geliştirme için daha uygundur.
* **Endüstri ve araştırma kullanım senaryoları:** PyTorch genellikle akademik çalışmalar için tercih edilirken, TensorFlow endüstri çözümleri için daha yaygın olarak kullanılmaktadır.

**8. Derin Öğrenmenin Uygulama Alanları**

**1. Bilgisayarlı Görü**

* **Görüntü sınıflandırma ve nesne tespiti:** Görüntülerdeki nesneleri tanıma ve etiketleme işlemleridir.
* **Semantic segmentation:** Görüntülerdeki her pikselin anlamlı bir etikete atanmasıdır.

**2. Doğal Dil İşleme**

* **Metin sınıflandırma ve duygu analizi:** Metinlerin anlamını çıkarmak ve duygusal içeriklerini analiz etmek.
* **Makine çevirisi ve metin üretimi:** Bir dildeki metni başka bir dile çevirmek ya da metin üretmek için kullanılan modellerdir.

**3. Oyun ve Simülasyon**

* **Reinforcement learning (RL) ve Deep Q-Networks (DQN):** Yapay zeka ajanlarının çevreleriyle etkileşime girerek en iyi eylem stratejisini öğrenmesini sağlar.