T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KNIME VERİ ANALİZ PLATFORMU KULLANARAK DERİN ÖĞRENME ALGORİTMALARI İLE İMGE SINIFLANDIRMA

Hilal ÇELİK

Yüksek Lisans Tezi

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

HAZİRAN, 2021

T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Bilgisayar Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

KNIME VERİ ANALİZ PLATFORMU KULLANARAK DERİN ÖĞRENME ALGORİTMALARI İLE İMGE SINIFLANDIRMA

Tez Yazarı
Hilal ÇELİK

Danışman
Doç. Dr. Ahmet ÇINAR

HAZİRAN, 2021
ELAZIĞ

T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Bilgisayar Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Başlığı:	Knime Veri Analitik Platformu Kullanarak Derin Öğrenme Algoritmaları ile Imge Sınıflandırma
Yazarı:	Hilal ÇELİK
İlk Teslim	08.06.2021
Savunma	08.07.2021

TEZ ONAYI

Fırat Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına göre hazırlanan bu tez aşağıda imzaları bulunan jüri üyeleri tarafından değerlendirilmiş ve akademik dinleyicilere açık yapılan savunma sonucunda OYBİRLİĞİ ile kabul edilmiştir.

		İmza
Danışman:	Doç. Dr. Ahmet ÇINAR Fırat Üniversitesi, Mühendislik Fakültesi	Onayladım
Başkan:	Prof. Dr. Ali KARCI İnönü Üniversitesi, Mühendislik Fakültesi	Onayladım
Üye:	Doç. Dr. Taner TUNCER Fırat Üniversitesi, Mühendislik Fakültesi	Onayladım

Bu tez, Enstitü Yönetim Kurulunun ...../...../20..... tarihli toplantısında tescillenmiştir.

İmza

Doç. Dr. Kürşat Esat ALYAMAÇ
Enstitü Müdürü

BEYAN

Fırat Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırladığım "Knime Veri Analitik Platformu Kullanarak Derin Öğrenme Algoritmaları ile Imge Sınıflandırma" yüksek lisans tezimin içindeki bütün bilgilerin doğru olduğunu, bilgilerin üretilmesi ve sunulmasında bilimsel etik kurallarına uygun davrandığımı, kullandığım bütün kaynakları atıf yaparak belirttiğimi, maddi ve manevi desteği olan tüm kurum/kuruluş ve kişileri belirttiğimi, burada sunduğum veri ve bilgileri unvan almak amacıyla daha önce hiçbir şekilde kullanmadığımı beyan ederim.

ELAZIĞ, 2021

Hilal ÇELİK

ÖNSÖZ

Bu çalışmada bilgi ve deneyimleriyle beni yönlendiren, desteklerini esirgemeyen ve her konuda yardımcı olmaya çalışan, tecrübelerini paylaşan değerli danışman hocam Doç. Dr Ahmet ÇINAR'a ve Öğretim Görevlisi Elif Işılay ÜNLÜ'ye teşekkürlerimi sunarım.

Hilal ÇELİK

ELAZIĞ, 2021

İÇİNDEKİLER

Sayfa

ÖZET

Knime Veri Analitik Platformu Kullanarak Derin Öğrenme Algoritmaları ile Imge Sınıflandırma

Hilal Çelik

Yüksek Lisans Tezi

FIRAT ÜNİVERSİTESİ
Fen Bilimleri Enstitüsü

BilgisayarMühendisliği Anabilim Dalı
June 2021, Sayfa xii+48

Çağımızda veri her yerde ama bu verilerin tamamı anlamlı ve işe yarar bilgi olarak bulunmuyor. Küreselleşen dünyada her birey ve her makine bir veri kaynağı ve sürekli artan büyük bir ivme ile veri üretmektedir. Cep telefonlarımız, güvenlik kameraları, banka kartları, akıllı arazi sulama sistemleri, kullanılan kimlik kartları, su sayaçları ve daha birçoğu veri üretiyor. Üretilen bu verilerden anlamlı ve işe yarar bilgi elde etmek çağımızın en büyük çalışma alanlarından birini oluşturmaktadır.

Son yıllarda hızlı gelişen veri bilimi, hayatı birçok açıdan kolaylaştıran yapay zeka teknolojisinden yaralanmaktadır. Sağlık, eğitim, sanayi, turizm yapay zekadan yaralanan alanların başında gelir. Yapay zekanın alt dalları olan makine öğrenmesi ve derin öğrenme algoritmaları sayesinde hayat daha kolay oluyor.

Çinde ortaya çıkan ve tüm dünyayı etkisi altına alan Covid-19 pandemi döneminde de yine yapay zekadan faydalanılan birçok sistem geliştirildi. Geliştirilen sistemler sayesinde bu zorlu süreç en az hasarla ve en kısa sürede atlatılmaya yöneliktir. Bu süreçte sürece olumlu etkisi olan alınabilecek tedbirler vardır. Bu alınabilecek önlemlerden biri maske takmaktır.

Bu tez çalışmasında insanların maske takıp takmadığını derin öğrenmeye dayanan yaklaşımlar kullanılarak Knime veri analiz programı kullanılarak tespit edilmeye çalışılmıştır. İnsanların maskeli ve maskesiz resimleri eğitilerek oluşturulan sistem sayesinde, test verisinden sisteme yeni giren verinin maskeli ya da maskesiz olduğu tespit edilmeye çalışılmıştır. Bu çalışmasında AlexNet ve Lenet derin öğrenme mimarilerinden faydalanılmıştır.

Anahtar Kelimeler: Resim sınıflandırma, Kollektif öğrenme, Derin öğrenme, Yapay zeka, Knime

ABSTRACT

Image Classification with Deep Learning Algorithms Using Knime Data Analytics Platform

Hilal Çelik

Master's Thesis

FIRAT UNIVERSITY
Graduate School of Natural and Applied Sciences

Department of Computer Engineering
June2021,Page xii+48

In our age, data is everywhere, but not all of this data is available as meaningful and useful information. In the globalizing world, every individual and every machine is a data source and generates data with an ever-increasing acceleration. Our mobile phones, security cameras, bank cards, smart land irrigation systems, used ID cards, water meters and many more generate data. Obtaining meaningful and useful information from these produced data constitutes one of the biggest working areas of our age.
In recent years, data science, which has developed rapidly, benefits from artificial intelligence technology that makes life easier in many ways. Health, education, industry and tourism are among the areas benefiting from artificial intelligence. Life is easier thanks to machine learning and deep learning algorithms, which are sub-branches of artificial intelligence.

During the Covid-19 pandemic that emerged in China and affected the whole world, many systems using artificial intelligence were also developed. Thanks to the developed systems, this challenging process is aimed to be overcome with the least damage and in the shortest time. In this process, there are measures that can be taken that have a positive effect on the process. One of the precautions that can be taken is to wear a mask.

In this thesis, it has been tried to determine whether people are wearing masks or not by using Knime data analysis program by using approaches based on deep learning. Thanks to the system created by training masked and unmasked images of people, it has been tried to determine whether the new data entering the system from the test data is masked or unmasked. In this study, AlexNet and Lenet deep learning architectures were used.

Keywords: Image classification, Collective learning, Deep learning, Artificial intelligence, Knime

ŞEKİLLER LİSTESİ

Sayfa

TABLOLAR LİSTESİ

Sayfa

KISALTMALAR

AI : Artificial Inteligence
ML : Machine Learning
NN : Neural Network
DL :Deep Learning
YSA : Yapay Sinir Ağları
CNN : Convolutional Neural Network
MAVL : Majority Voting
Bagging : BootStrap Aggregation
ReLU : Rectified Linear Unit
MLP : MultiLayer Perceptron
RBM : Restricted Boltzmann Machines(Kısıtlı Boltzmann Makineleri)
DBM : Derin İnanç Ağları(Deep Belief Network)
DeepMLP : Deep Multi Layer Perceptron
SimpleMLP : (Basit Multi Layer Perceptron)
YSA : Yapay Sinir Ağı
GPU : Graphics Processing Unit(Grafik İşlemci Birimi)
JVM : Java Virtual Machine
DL4J : Deeplearner4J
API : Application Programming Interface(Uygulama Programlama Arayüzü)
JVM :JavaVirtual Machine

1. GİRİŞ

Veriden anlamlı ve işe yarar bilgiyi elde etmek için hali hazırda birçok yöntem vardır. Bu yöntemlerin başında Makine öğrenmesi algoritmaları ve derin öğrenme yöntemleri gelir. Fakat aynı algoritma farklı veri setleri üzerinde çalıştırıldığında farklı sonuçlar vermektedir. "Bütün kendini oluşturan parçalardan büyüktür" felsefesinden yola çıkarak bu çalışmada bir çok algoritmadan aynı anda toplu olarak yararlanmaya çalışılmıştır. Hem zamandan ki son yıllarda zamandan kazanç çok daha önem kazanmıştır, hem de test verileri doğruya daha yakın olma olasılığını arttırıyoruz.

Sisteme giren örneklerin en yüksük oranda sınıflandırılması amaçlandığından bu tez çalışmasında algoritmaların çoğunluğun gücünden yaralanılarak örnekler tek başına çalıştırılan algoritmalara oranla çok daha iyi sonuçlarla sınıflandırılmıştır. Literatürlerdeki benzer çalışmalarda Knime'de kolektif derin öğreneme uygulaması bulunmamaktadır.

2. VERİ BİLİMİNE GİRİŞ

Veri; ham(işlenmemiş) bilgilere denir. Veriler her zaman hayatımızda işlenmeye hazır ve işe yarar halde değildir. Veri işlenmeye hazır hale getirilinceye kadar birçok aşamadan geçer. Bu aşamalar sonunda işlenmeye hazır hala gelen veri artık üzerinde çalışıp anlamlı ve işe yarar sonuçlar elde edilmesi mümkün olur. Tabi her veri işe yarar veri değildir. Veri madenciliği yapılacak keşif için doğru veriyi biriktirmekle başlar.

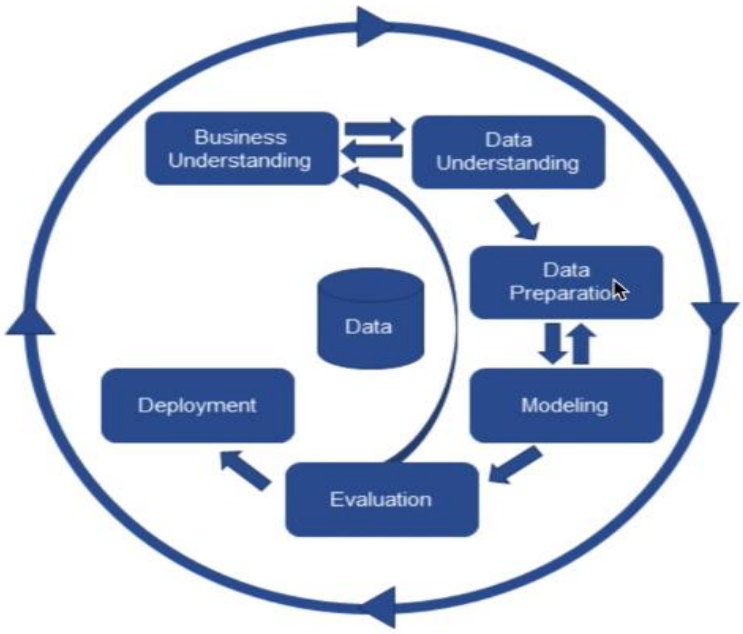
Verinin gücünden yararlanmak için veriden anlamlı ve işe yarar bilgiyi elde etmek gerekir. Çağımızda veri her yerde ama bu verilerin tamamı anlamlı ve işe yarar bilgi olarak bulunmuyor. Son yıllarda her makine bir veri kaynağı ve sürekli artan büyük bir ivme ile veri üretmektedir. Elimizdeki telefonlar, güvenlik kameraları, banka kartları, akıllı arazi sulama sistemleri, kullanılan kimlik kartları, su sayaçları ve daha birçoğu veri üretiyor. Üretilen bu verilerden anlamlı ve işe yarar bilgi elde etmek çağımızın en büyük çalışma alanlarından birini oluşturmaktadır. Bu çalışma alanında veri bir serüvenin içine sürüklenmektedir. Bu serüvende üretilen verileri, veri tabanlarında toplamak, ihtiyaç duyulduğunda istenen yerlere iletmek ve işlemek çalışma alanımızın temel amacıdır.

2.1. Veri Bilimi Yöntemleri

Veriyi işlenmeye hazır hale getirmek için veri bilimi yöntemleri kullanılır. Veri işlenirken kullanılan birçok veri bilimi yöntemi vardır. Veri bilimi yöntemi seçilirken üzerinde çalışılan projenin optimum verimle sonuçlanmasını sağlayan veri bilimi yöntemi seçilmelidir. Veri ham halden işlenmeye hazır hale getirilmeye çalışılırken kullanılan veri bilimi yöntemlerinden sık kullanılandan bazıları CRISP-DM, SEMMA ve KDD'dır. Bu veri bilimi yöntemlerinin dışında şirketlere özel kullanılan veri bilimi yöntemlerde bulunmaktadır. Bu yöntemler veri madenciliği sürecinin nasıl yapılması gerektiği ifade etmektedir.

2.1.1.CRISP-DM (Cross Industry Standard Process for Data Mining)

Veri madenciliği (CRISP-DM), bir yinelemeli süreç modelidir. 1996 ve o zamandan beri en çok tercih edilen veri bilimi yöntemi CRISP-DM şu özeliklerden oluşur: iş anlayış; verilerin anlaşılması; veri hazırlama; modelleme; değerlendirme; ve dağıtım [1].



Şekil 2.1. CRISP-DM döngüsü [1]

Ele aldığımız veri bilimi yöntemlerinden en çok kullanılanlarından birisi CRISP-DM' dır.

Şekil 2.1'de görüldüğü üzere bazı adımlar arasında döngüler mümkündür ve bunların proje yönetimi açısından kolaylık sağladığı için önemi büyüktür. Temel olarak 6 adımdan oluşan bu veri bilimi yönteminde projeye başlarken öncelikle problemin tanınması ve ikinci olarakverinin tanınmasıveya probleme uygun verinin toplanması şeklinde iki temel adımla başlar. Üçüncü adım verinin hazırlanması evresine geçilir. Bu aşamada veri üzerinde kirli ve eksik veri gibi veriler üzerinde çalışılır. Dördüncü adımda modelin inşasına geçilir ve beşinci adımda ise modellerin sonuçları değerlendirilir ve gerekirse iyileştirmeler yapılır. Altıncı ve son adım ise raporlama adımı; modelin analist ve son kullanıcılara sunulduğu ve değerlendirme sonucu alınan adımdır. Eğer bu aşama sonucunda istenilen derecede bir başarı oranı elde edildiyse yada değerlendirme olumlu geçtiyse proje sonlandırılır ve üretime geçilir [1,2].

2.2. Problemi ve Veri Tanımak

Veri madenciliğinin veriden bilgiye ulaşma serüveninde problem ve veri arasında tutarlı bir ilişki olmalıdır. Veri setimiz çözmeye çalıştığımız problemimize uygun olmalıdır. Örneğin; ilkokuldaki çocuklar üzerinde bir problemin kaynağının araştırılması, çözülmesi veya var olan sorunun iyileştirilmesi gibi bir amacımız var ise veri setimizde araştırılan yaş aralığını kapsayan bir veri seti olması uygun olacaktır.

2.2.1.Problemi Tanımak

Bir projeye başlarken ilk yapılaması gereken şey problemin ne olduğunu anlamak esastır. “Bu projenin ihtiyacı ve hedefi nedir? Hedef kitlenin ihtiyaçları nelerdir? “ gibi soruların cevapları aranır.

CRISP-DM veri bilimi yönteminin 1. adımı burada uygulayarak projenin hedeflerini ve ihtiyaçlarını karar verilir. Kullandığımız veri setinden yola çıkarak hedefimizi belirlemeye çalışıyoruz.

2.2.2. Veriyi Tanımak

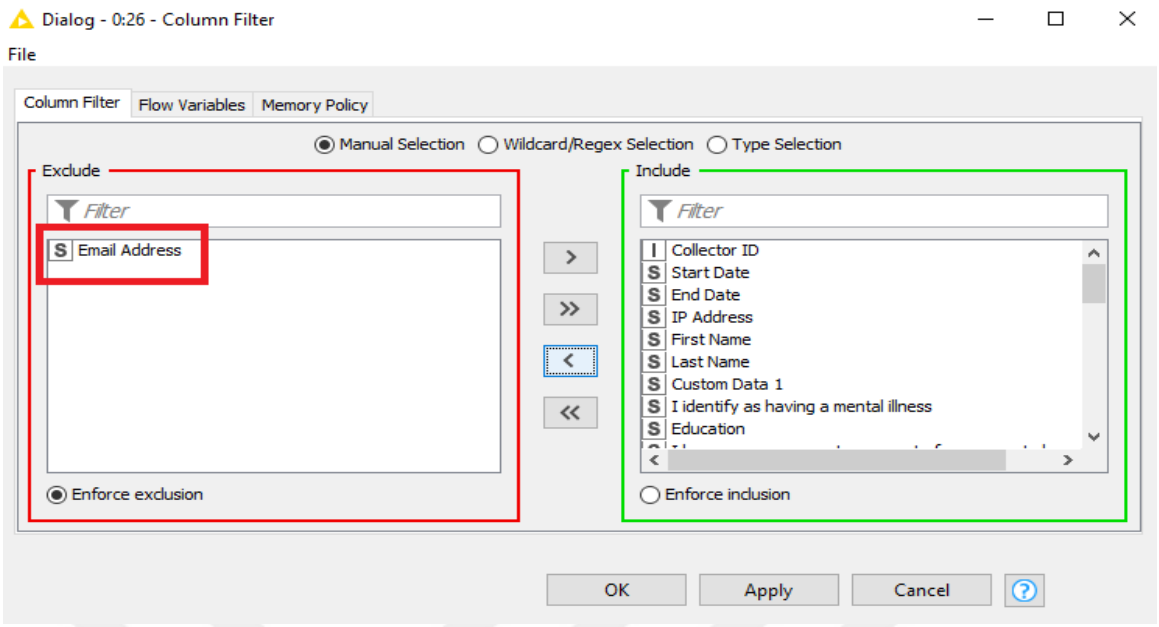
CRISP-DM veri bilimi yönteminin 2. adımı veriyi tanımaktır. Verilerini tanımaya çalıştığımız veri setinde veri kolonları yol gösterecektir. Kullanılan verilerin sayısal mi sözel mi, resim mi gibi kriterler incelenir. Veriyi daha iyi yorumlamak için Knime’de “Scatter Plot” gibi düğümler kullanılabilir. Böylece veri daha kolay yorumlanabilecek.

2.3. Veriyi Hazırlama

CRISP-DM veri bilimi yönteminin 3.adımı veriyi işlenmeye hazır hale getirmektir. Veri setimizin eksiklikleri giderilmesidir. “ Hangi kolonlar üzerinde çalışılacak? Eksik verilere ne gibi işlemler uygulanacak?” gibi sorunların çözümü CRISP-DM veri biliminin 3. adımında gerçekleştirilir.

Verinin işlenmesi, eksik verilerin bulunması ve eksik veriye nasıl bir uygulama yapılacağına karar verilir. Eksik veri sabit bir değer mi almalı yoksa bulunduğu kolonun ortak değerini mi almalı yada kullanılmayan kolonlar veri setinden çıkarılmalı gibi kararlar verilir ve uygulanır. Knime’nin sunduğu düğümlerden 3. adımda işimize en çok yarayan düğümler seçilir.

“Colon Filter” düğümü; veri setinde işimize yaramayan ve veri setinin anlaşılmasını zorlaştıran kolonlar çıkartılabilir. Örneğin “yeni müşteriye kredi verilebilir mi?” gibi bir çalışmada müşterilerin e-mail adresi çıkartılması veri kalabalığını önler. Çünkü kişilerin email adresine bakarak müşteriye kredi verilmesi arasında bir bağıntı yoktur. Bu düşüncemizden dolayı “Colon Filter” düğümü yardımıyla “Email Address” kolonunu veri setimizden ayırıyoruz. Şekil 2.2’de Knime’de “colon fitler” kullanımı gösterilmiştir.



Şekil 2.2. Knime'de "colon fitler" kullanımı

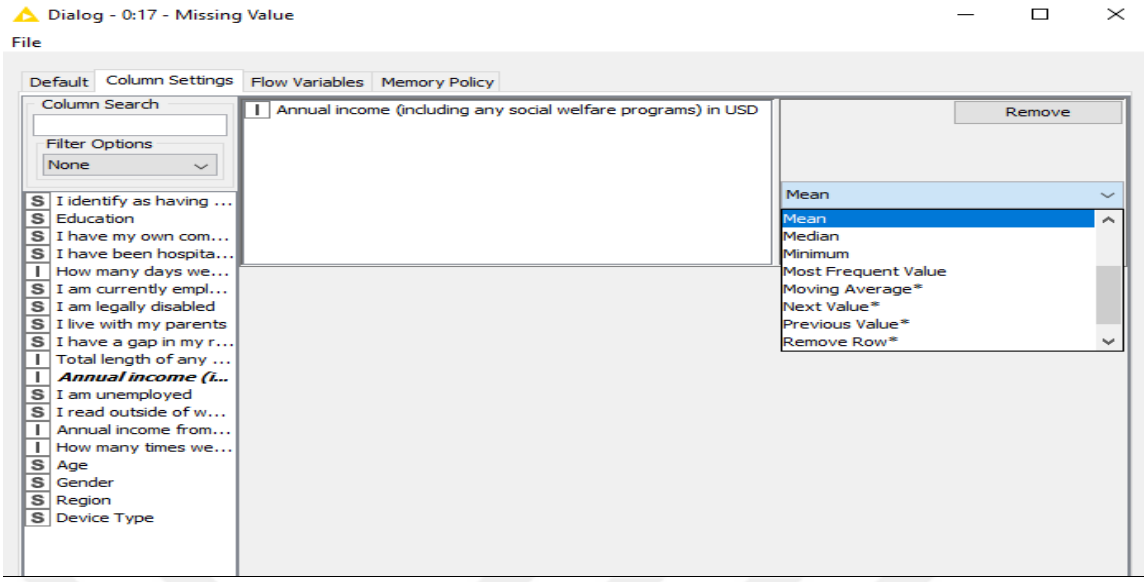
2.4. Veriyi Temizlemek

Veriler işlenirken sorunlarımızdan bir tanesi gürültülü, eksik ve tutarsız veriler üzerinde çalışmaktır. Veri setinin bütünlüğünü bozmamak buradaki çalışmada esas konudur. Veri bütünlüğü bozulmadan veri setindeki eksik alanlar doldurulmalıdır. Gerekirse uzun veriler kısaltılmalı, sayısal ve kelime tipinde olan veri alanları, yapılan çalışmaya uygun olarak dönüştürülebilir.

2.4.1. Eksik Veri

Veri setinde olmayan verilere eksik veri (missng value) denir. Knime da eksik veriler "?" ile gösterilir. Verimizin string ve integer olması durumlarında knime farklı alternatifler sunar. Knime'nin eksik veriler için sunduğu en basit çözüm yolu "ignore missing value" seçeneğidir. Basit olarak "ignore missing value" seçilerek göz ardı edilebilir yani bu eksik veri olan hücreler dikkate alınmaz. Bunun dışında eksik veriler için yapılabilecek birçok alternatif mevcuttur. Bunlardan bazıları; veri seti kolonun ortalamasının almak, kolondaki en büyük ya da en küçük değeri atamak olabilir.

Veri setimizde eksik veri sahibi olan kolonumuz için Knime'nin sunduğu seçeneklerden birini veri setinin doğruluğuna en yakın değeri verecek olan alternatif seçilebilir. Şekil 2.3'de gösterilen bir veri seti örneğinde kişilerin maaşları için ortalama değer girilmesini uygun olacaktır.



Şekil 2.3. Missing Value düğümü iç yapısı

2.4.2. Gürültülü Veri

Gürültülü veri; verilerin hatalı ya da aykırı olması durumudur. "Gürültü nedir?" Gürültü, ölçülen bir değişkendeki rastgele bir hatadır [3].

Örnek: maaş=-300tl, yaş=-54 yaş veya maaş gibi veriler (–) değer alamaz.

2.4.3. Tutarsız Veri

Veri ve verinin kaynağı kıyaslandığında ortaya bir yanlış çıkması durumudur. Örneğin yaşı 21 olan kişinin doğum tarihi 12.01.1999 olmaz.

2.5. Öğrenme Türleri

Verilerden anlamlı ve işe yarar bilgilere ulaşma esnasında her veri kedine benzeyen diğer veriler ile aynı yerde bulunmalıdır. Bu sayede verilere daha kolay ulaşılır ve daha kolay işlenir. Veriler işlenirken önce bir sistem oluşturulur. Bu sistem önce problemi ve problemin çözümüne giden yolu öğrenmesi gerekir. Oluşturulan sistemin bulduğu bu çözüm yolunu karşılaştığı yeni problemlerde uygulayabilir olmalı. Sistem oluşturulurken ilk aşamada sistem eğitilir. Bu eğitim işlemi yapılırken veri setinin depolanma şekline göre öğrenme tip seçilir.

2.5.1 Gözetimli Öğrenme(Supervised Learning)

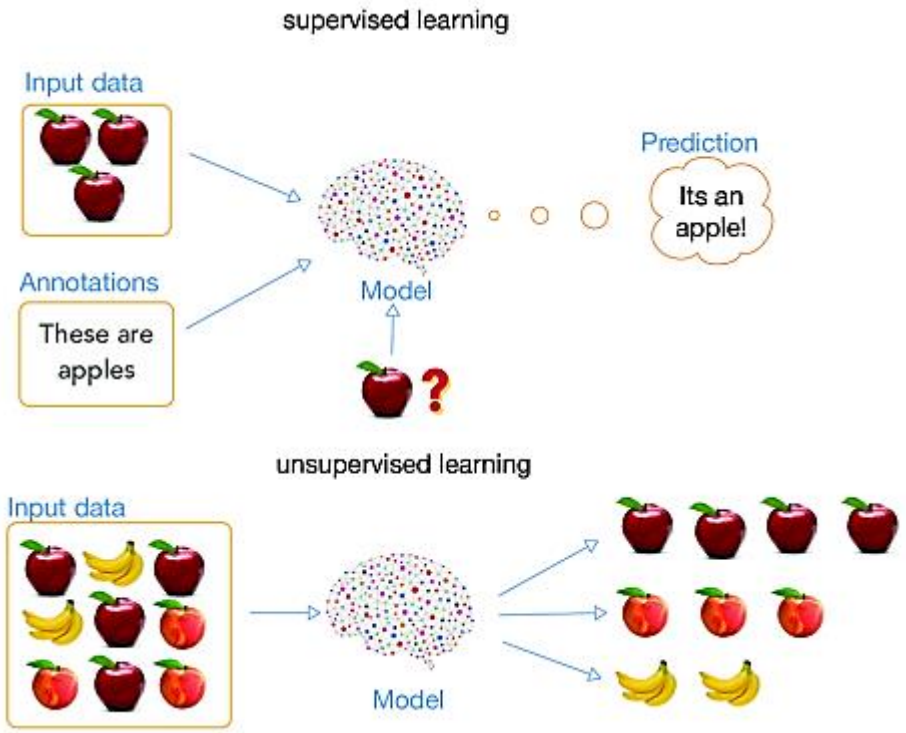
Gözetimli öğrenmede sisteme sürekli bilgi sağlanır. Sistemde hangi sınıfların var olduğu ve bu sınıfların hangi kolon bilgisine sahip olduğu önceden bellidir. Hangi veri hangi sınıfa dahildir

önceden bilinir. Bu tezde gözetimli öğrenme yöntemi kullanılmıştır. Dolayısıyla bu çalışmada kullanılan veri setinde sınıflar ve hangi verinin hangi sınıfa dahil olduğu önceden bellidir [4]. Bu tez çalışmasında Gözetimli Öğrenme(Supervised Learning) yöntemi kullanılmıştır. www.kaggle.com sitesinden "With/Without Mask Mask Detection for COVID-19" veri seti indirilmiştir. Bu veri seti verileri iki ayrı sınıfta ve hangi veri hangi sınıfa dahil olduğu önceden belli olarak sisteme yüklenmiştir. Sistem bu veriler üzerinde çalışarak sistemi eğitecek ve daha sonra gelecek verinin hangi sınıfa dahil olacağını tahmin etmeye çalışır.

2.5.2. Gözetimsiz Öğrenme(Unsupervised Learning)

Gözetimsiz öğrenmede sisteme gelecek veri hakkında sistem hangi verinin geleceği bilgisine önceden sahibi değildir. Sistemde kaç sınıf olduğu, gelen verilerin hangi sınıfta dahil olduğu bilgisi belli değildir. Sınıf sayısı ve verilerin hangi sınıfa dahil edileceğini makine kendisi öğrenmeye çalışarak karar verir. Gözetimsiz öğrenmeye verebileceğimiz en iyi bir örnek olarak mantık sorularıdır. Daha önce karşılaşmadığımız ilk bakışta ne sorusu olduğunu anlamakta zorluk çektiğimiz ama biraz düşündükten sonra matematik bilgisi ile çözülebileceğine karar vermek gibi [4].

Eğer bir algoritma hem gözetimli hem de gözetimsiz öğrenmeyi kullanırsa buna yarı gözetimli öğrenme(Semi-supervised Learning) algoritması denir [4]. Gözetimli ve gözetimsiz öğrenme arasındaki fark şekil 2.4.'de gösterilmiştir. Denetimli öğrenmede eğitim kümesinin örnekleri sisteme girmeden önce hangi sınıfa dahil olduğu belli iken denetimsiz öğrenmede ise örneğin hangi sınıfa dahil olduğu önceden belli değildir. Denetimli öğrenmede sistemdeki benzer örnekler aynı sınıfta yer alır ve algoritma daha çok ezber yapan bir insan gibi çalışır. Diğer bir deyişle bir çocuğa "elma" öğretilecekse, birçok kere ve farklı şekillerde elma gösterilir ve defalarca görerek çocuk elmayı öğrenir. Denetimsiz de ise sadece elma değil elmanın yanında birçok farklı meyve de gösterilerek öğretilir. Doğada varlıkların sınıflandırılması göz önünde bulundurulduğunda denetimsiz öğrenmenin daha çok kullanılması gerektiği anlaşılır. Çünkü doğadaki varlıklar kategorik olarak bulunmaz. Bir yöne baktığımızda bile aynı karede birden fazla kategoride varlık görebiliriz. Bu varlıkları birbirinden ayırmamızı sağlayan şey aslında denetimsiz öğrenmedir.



Şekil 2.4. Gözetimli ve Gözetimsiz Öğrenme [5]

3. YAPAY ZEKÂ

Yapay zekâ insanının düşünme şekline yakın çalışan bir sistemdir. Yapay zekâ, insan davranışlarının makine tarafından taklit edilmesini sağlamaktadır [6]. Yapay zekâ (AI), insanlar gibi çalışan ve tepki veren akıllı makinelerin yaratılmasını vurgulayan bir bilgisayar bilimi alanıdır. Belirli bir görevi gerçekleştirmek için yazılım rutinlerini belirli bir dizi talimatla elle kodlamak yerine, makine, görevi nasıl gerçekleştireceğini öğrenme yeteneği veren büyük miktarda veri ve algoritmalar kullanılarak "eğitilir" [7].

Yapay zekâ insan gibi düşünmez aslında insan gibi düşündürülür. Yani kendisini oluşturan komutların ötesine gidemez. Diğer bir deyişle makineler onları oluşturan programların komutlarını icra eder. Bu program komutları doğadaki canlılardan en mükemmeli olan insanı baz alarak oluşturuluyor. Makineler normal bir insan zekâsının yapabildiği bir hesaplamayı insana oranla çok kısa bir sürede gerçekleştirebilirken, karar verme aşamasında bir insan kadar karmaşık karar veremez. Daha öncede de belirtildiği gibi kendini oluşturan programın ötesine çıkamaz. Çünkü genelde makineler belli bir görevi yerine getirmek için programlanır. Yani her makineler kendine has bir görevi vardır. Örnek vermek gerekirse bir akılı süpürge makinesi ve hesap makinesinin yaptığı işler birbirinden farklıdır ama insan basit seviyede ikisinin de yaptığı işi yapabilir.

3.1. Lojistik Regresyon

Bir sınıflandırma algoritması olan lojistik makine öğreniminin kapsamına girmektedir. Lojistik regresyon genellikle doğrusal sınıflandırma problemlerinde kullanılır. Lojistik regresyonu öğrenmek derin öğrenmeyi anlamayı kolaylaştırır. Çünkü derin öğrenmede birden fazla katman söz konusu iken lojistik regresyonda katman yoktur. Tek sinir hücreli ve aynı zamanda tek katmanlı bir YSA'dır. Sadece girdi parametreleri ve her parametre ile çarpılan ağırlık değerlerinin toplanmasına eklenen bias değerinin sonucu çıktısını oluşturur. Derin öğrenmede ise buna ek olarak gizli katmanlar vardır. Derin öğrenme lojistik regresyonun birden fazla tekrar edilmesi gibi düşünülebilir. İşte buradaki birden sonraki katmanlar gizli katmaları oluşturur.

3.2. Yapay Sinir Ağıları(Artificial Neural Networks)

ANN derin öğrenmenin bir alt koludur. Yani ANN'nın olduğu yerde derin öğrenme vardır.

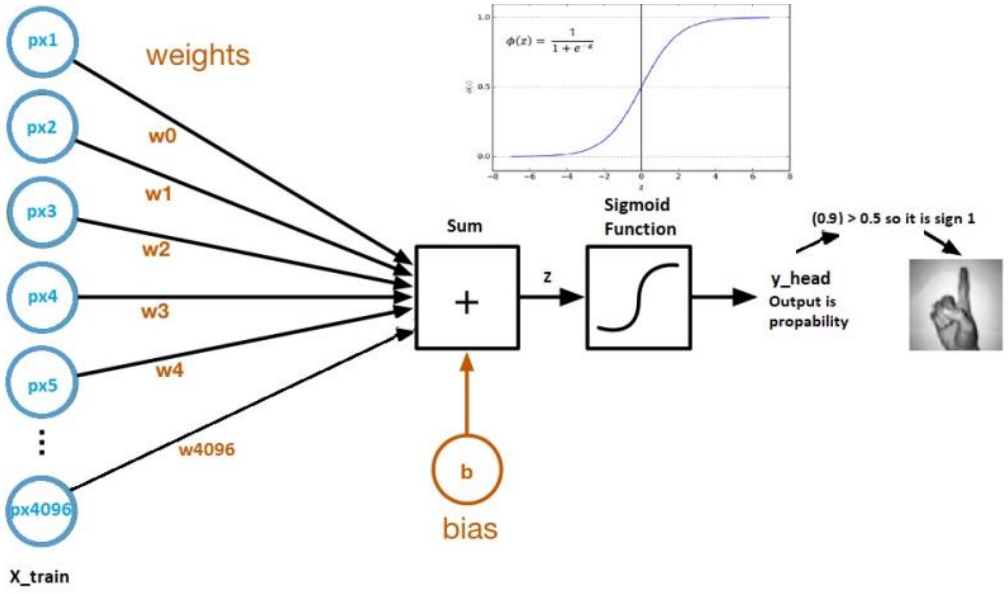
Makineler çoğu zaman insanlar gibi davranabilir ve ancak dünyayla ilgili bol miktarda bilgiye sahiplerse tepki verebilirler. Makinelerde sağduyu, muhakeme ve problem çözme gücünü başlatmak zor ve sıkıcı bir yaklaşımdır. Makine öğrenimi, yapay zekânın başka bir temel

parçasıdır. Herhangi bir denetim olmadan öğrenme, girdi akışlarındaki kalıpları belirleme becerisi gerektirirken, yeterli denetimle öğrenme, sınıflandırma ve sayısal regresyonları içerir [7].

Yapay sinir ağları biyolojik sinir sistemlerinden faydalanarak tasarlanmıştır. Biyolojik sinir hücreleri gibi synapsler araılığıyla ile iletişim sağlarlar ve axon'lar vasıyatsıyla bir sinir hücresinin işlediği bilgiler diğer hücrelere aktarılır. Bu yapıdan ilham ılınarak geliştirilen YSA benzer şekilde yapay sinir hücreleri dışarıdan gelen bilgileri bir toplama fonksiyonu ile toplar ve aktivasyon fonksiyonundan geçirerek çıktıyı üretip ağın üzerinden diğer hücrelere (proses elemanlarına) gönderir. YSA' yı birbirlerine bağlayan bağlantıların değerlerine ağırlık değerleri denmektedir. Proses elemanları birbirlerine paralel katmanların bir araya gelerek bir ağ oluştururlar. Bu katmanlar; girdi katmanı, ara katmanlar ve çıktı katmanı olmak üzere 3katmandan oluşmuştur [8].

İnsan sinir hücresinden esinlenerek sinir ağları ve devamı olan derin öğrenme teknolojileri ortaya çıkmıştır.

Şekil 3.1'de Logistik regresyonu temel alan YSA'nın basit matematiksel gösterimidir. Şekilde görülen pixeller herhangi bir resmin pixellerini temsil etmektedir. Bu şekilde kullanılan pixel sayısı, 64*64'lük resim için bu çarpım sonucu olan 4096 pixel olur. Her pixel bir ağırlık(w) değeri ile çarpılarak toplanır ve bias eklenerek bir sonuç elde edilir. Elde edilen sonuç seçilen aktivasyon fonksiyonunun eşik değerini geçip geçmediğine bakılır. Şekil 2.3'de şeçilen aktivasyon fonksiyonu sigmoid aktivasyon fonksiyonudur. Yine burada seçilen eşik değeri 0.5 tir. Yani bulunan sonuç(z) değeri 0.5'ten büyükse, eşik değerini geçtiği için sonuç doğru ve z değeri 0.5'ten küçükse eşik değerini geçmediği için sonuç yanlış kabul edilir.



Şekil 3.1. Bir nöronun (sinir hücresi) matematiksel modeli [9]

Yapay sinir ağının matematiksel gösterimin denklem(computation graf) şekil 3.1'de gösterilmiştir.

$$z = b + px1w1 + px2w2 + ... + px4096*w4096 \quad (3.1)$$

y: x'in değerine bağlı olduğundan bağımlı değişkendir. Girdiye ait skoru verir.

x: bağımsız değişken, girdi.

W: ağırlık parametresi diğer bir deyişle her pixelin katsayısı, yani her bir pixelin ne kadar öneme sahip olduğudur.

z değerini Sigmoid fonksiyona verip hesaplama yapıyoruz. Sigmoid fonksiyonu: Verdiğimiz z değerini 0 ile 1 arasında bir değere eşitler. Sigmoid func. türevi alınabilen bir fonksiyondur. Türevi alınabilir olması sayesinde weight ile bias 'ı güncelleyebiliyoruz [9].

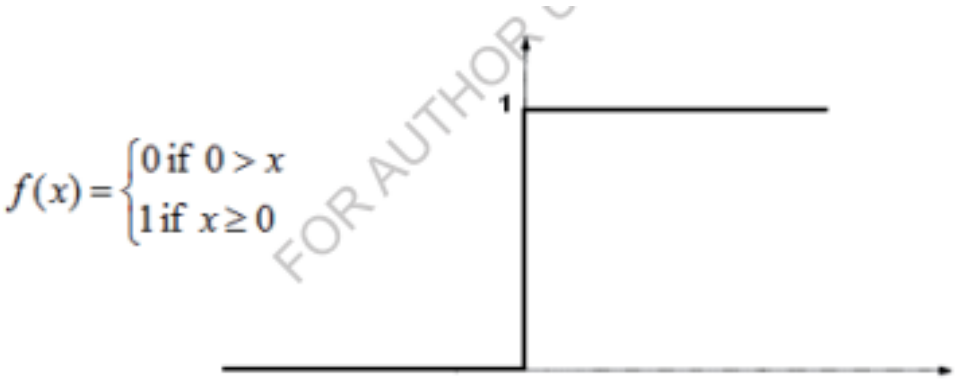
3.2.1. Weight ve Bias Güncelleme

Geriye doğru güncellemeler maliyet(cost) fonksiyonunu denklem 3.2'de gösterildiği gibi ağırlığa göre türevi alınarak eğim bulunur. Bulunan eğim mevcut ağırlıktan çıkarılır. Böylece ağırlık güncellenmiş olur. Ağırlıkların güncellenmesi sayesinde yapay zekâda öğrenme gerçekleşir. Eğim ise bir fonksiyonun bir noktaya göre türevidir. Bu güncelleme işlemi taki eğim sıfıra eşit oluncaya kadar yapılır

$$w = w - \alpha * \frac{\partial L}{\partial w} \quad (3.2)$$

3.2.2. Aktivasyon Fonksiyonu(Activation Function)

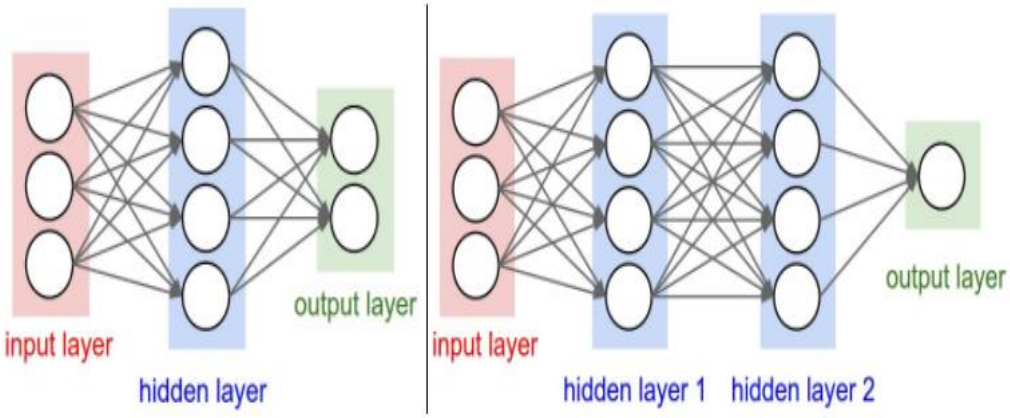
Her girdi(xi) kendine ait ağırlık(w) ile çarpılarak toplam z değeri hesaplanır ve bu değer bias ile toplanır. Bu aşamada devreye aktivasyon fonksiyonu girer ve istemin aktif olup olmayacağına kara verir [11]. Şekil 2.3 üzerinden aktivasyon fonksiyonunu anlatmak gerekirse; resmin her pixeli bir girdi oluşturur. Bu girdiler ağırlıklarla çarpılır ve bias eklenerek z değeri bulunur. Bulunan bu değer 0-1 aralığına aktivasyon fonksiyonu sayesinde indirgenir. Birçok aktivasyon fonksiyonu çeşidi vardır. Bu aktivasyon fonksiyonlarının en ilkel olanı basamak fonksiyonu Şekil 3.2.'de gösterilmiştir. Şekilden de anlaşıldığı gibi bulunan sonuç sıfır dan küçükse sıfır sıfırdan büyük veya sıfıra eşitse sonuç bir değerini alır. Tabi her aktivasyon değeri 0-1 aralında değildir. Farklı değerler alan aktivasyon fonksiyonları vardır.



Şekil 3.2. Adım Aktivasyon Fonksiyonu [12]

3.2.3. Sinir Ağlarında Katman Kavramı

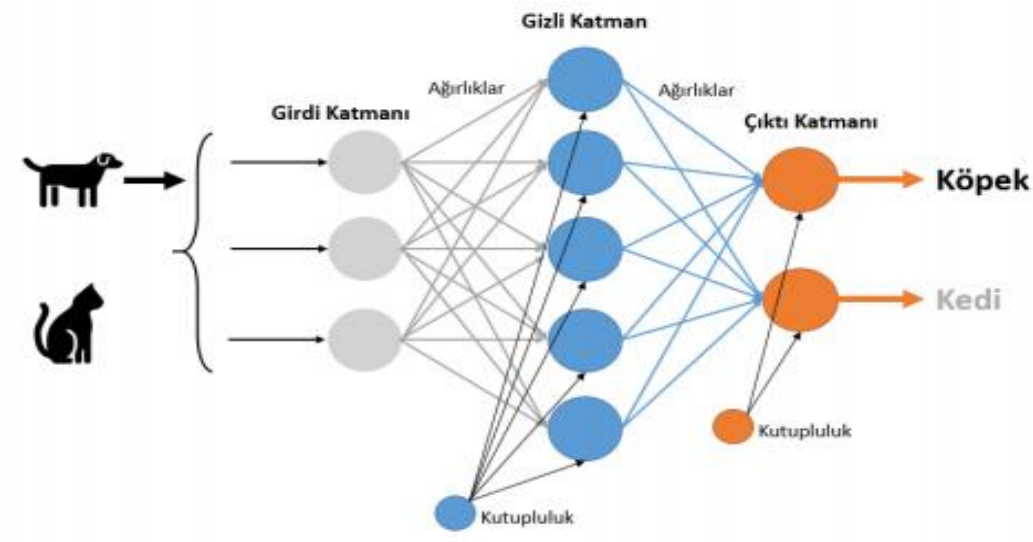
En az bir tane gizli katman içeren YSA'lar Derin Yapay Sinir Ağı olarak adlandırılır. Şekil 3.3.'de bir ve iki gizli katmanlı sinir ağları gösterilmiştir. Burada katman kavramı şöyle açıklanabilir; örneğin sistem giren bir resim için her katmanda resmin daha derinlemesine özellikleri oluşturulur. En basit haliyle fotoğraf çekmek için kullandığımız kameraların megapixel değerinin artması, derin öğrenmede katman sayınsın artması gibidir. Resim her katmanda daha da belirginleşir. Diğer bir deyişle resim her katmanda kaba şekilden daha ince detaylarına kadar özellik çıkarımı yapılır. Böylece eğitim veri seti çok daha iyi oranda eğitilir yani sisteme giren eğer bir resimse onu diğer resimlerden daha kolay bir şekilde ayırt eder. Bununla beraber test verisin doğruluk oranı arttırılır. Ayrıca gizli katman sayısı ne kadar fazla olur işlem süresi de bu süreyle doğru orantılı olarak artar.



Şekil 3.3. Tek Gizli Katmanlı ve Çok Katmanlı Sinir Ağı Yapısı [13]

3.2.4. Yapay Sinir Ağının Çalışma Prensibi

YSA çalışma prensibi olarak şekil 3.4.'de bir gözetimli öğrenme üzerinden açıklamak gerekirse; kedi ve köpek sınıfları girdi olarak sisteme girer. Eğitim verisi olan bu sınıflardaki her girdi ağırlıklarla çarpılır ve her sinir hücresi aktivasyon fonksiyonu aşar yada bu fonksiyona denk gelirse sinir hücresi aktif olur. YSA eğitim verisini eğitirken her veri için girdileri ağırlık ile çarpar ve aktivasyon değeri ile karşılaştırır. Bir sonraki veriyi eğitirken önceki veriden yararlanarak ağılıkları ve aktivasyon eşik değerini günceller. Bu güncellemenin amacı eğitim veri seti eğitilirken gerçeğe olabilecek kadar yakın tahminde bulunmaktır.



Şekil 3.4. YSA'da Sınflandırma Örneği [14]

3.2.5. Maliyet(Hata) Fonksiyonu

Bu fonksiyon, ağın ürettiği sonuç ile gerçek değer arasındaki farkın değerlendirmesi olarak tanımlanabilir [14]. Maliyet fonksiyonu denklem 3.3'de gösterildiği gibi hesaplanmaktadır. Buradaki amaç gerçek y değeri ile tahmin edilen y değeri minimuma indirmeyi sağlamak için ağırlıklar(w) güncellenerek hesaplanır. Böylece yapılan tahmin değerleri gerçeğe daha yakın olması sağlanır. Sisteme giren her resim için ayrı ayrı kayıp(lost) fonksiyonu yanı doğru tahmine olan yakınlığı hesaplanır. Hesaplanan her değer toplanarak toplam maliyet(hata) fonksiyonu fonksiyonu oluşturulur. Eğer bu hata fonksiyonu büyük çıkarsa ağırlık değerleri, bias güncellenerek yeniden model oluşturulur.

$$c(w) = \sum_{i=1}^{m} (y^i - f(x^i))^2 \tag{3.3}$$

Burada model güncellemesine karar veren eğittiğimiz veri setinde; model eğitilirken aktivasyon fonksiyonunda çıkan sonuca göre resme karar verilir. Aktivasyon fonksiyonundan

çıkan y_head değerine göre karar verilir. Burada bu değer 0.5'in altında ise 0 resmi ve 0.5 ve üzeri bir değerse 1 resmi bulunmuş olur. Gerçek resim ile bulunan değer arasındaki fark alınır. Bu fark büyükse model maliyet fonksiyonunu azaltacak şekilde güncellenir.

3.2.6. Öğrenme Oranı(Learnin Rate)

Ağın katman sayısı, katmanlardaki nöron sayısı, iterasyon sayısı, öğrenme oranı, momentum katsayısı, aktivasyon fonksiyonu, normalizasyon yöntemi, başlangıç ağırlıkları gibi parametrelerin değiştirilerek ağın eğitilmesi sağlanabilmekte ve eğitilen ağ test edilerek ağın performans ölçümü yapılabilmektedir. Tecrübeler $0.01 \leq \eta \leq 0.9$ aralığında seçilen öğrenme oranını iyi sonuçlar verdiğini göstermektedir[16].

Geriye doğru güncellemeler maliyet(cost) fonksiyonunu ağırlığa göre türevi alınarak eğim bulunur. Bu eğim öğrenme katsayısı ile çarpılır. Bulunan eğim mevcut ağırlıktan çıkarılır ve böylece ağırlık güncellenmiş olur. Eğim ise bir fonksiyonun bir noktaya göre türevidir. Bu güncelleme işlemi taki eğim sıfıra eşit oluncaya kadar yapılır. Öğrenme oranı çok büyük seçilirse avantaj olarak sistem hızlı çalışır ama dezavantaj olarak hata yüzeyinde büyük geçişler oluşur. Bu durumdan kaynaklı öğrenmenin gerçekleşeceği küçük alanlar atlanabilir. Küçük seçilen öğrenme katsayısı ise öğrenme süresini uzatır ve sistemin çalışması çok zaman alabilir.

3.3. Makine Öğrenmesi(Machine Learning)

Makineler insanların öğrettiği dibi düşünür. Yani onları oluşturan programların komutlarını icra ederler. Bu demek oluyor ki makineler bizim istediğimiz işi yapıyor. Hayatın hemen hemen her alanında kullanılan makineler insan beyninin öğrenme ve düşünme tarzından yola çıkarak programlanmaktadır. Makineler tıpkı insanın öğrendiği öğrenir. Örneğin insanda cinsiyet farkını öğrenmeye çalışan bir çocuğa sadece babası ve annesini görmek cinsiyet ayrımını yaptırmaz. Bunun yerine annesini, halasını, teyzesini vs gibi kadınları ve diğer yandan babasını, amcasını, dayısını vs görerek daha sonra karşısına çıkan başka birinin de cinsiyetin kendi ayırt edebilir. Basit haliyle anlatma çalıştığımız insanın öğrenmesi yapay zekânın eğitim veri setinin oluşturulmasıdır. Aynı şekilde insanın bu öğrenmeden sonra karşısına çıkan birisinin cinsiyetine karar vermesi yapay zekâda test veri setinin sonuçlarını temsil eder.

ML geniş kapsamlı uygulamalarla bilgisayar biliminin en hızlı büyüyen alanlarından biridir [17]. İstatistik ve matematikten faydalanarak verilerden çıkarım yapmayı sağlayan yapay zekâ teknolojilerinin bir alt dalıdır. . Özünde, verilerden öğrenerek karmaşık işlevleri tahmin eden bir dizi istatistiksel aracın temelini oluşturur. Makine öğrenimi alanı, verilere dayalı tahminler yapan algoritmaların oluşturulmasına odaklanmıştır [18].

3.4. Derin Öğrenme

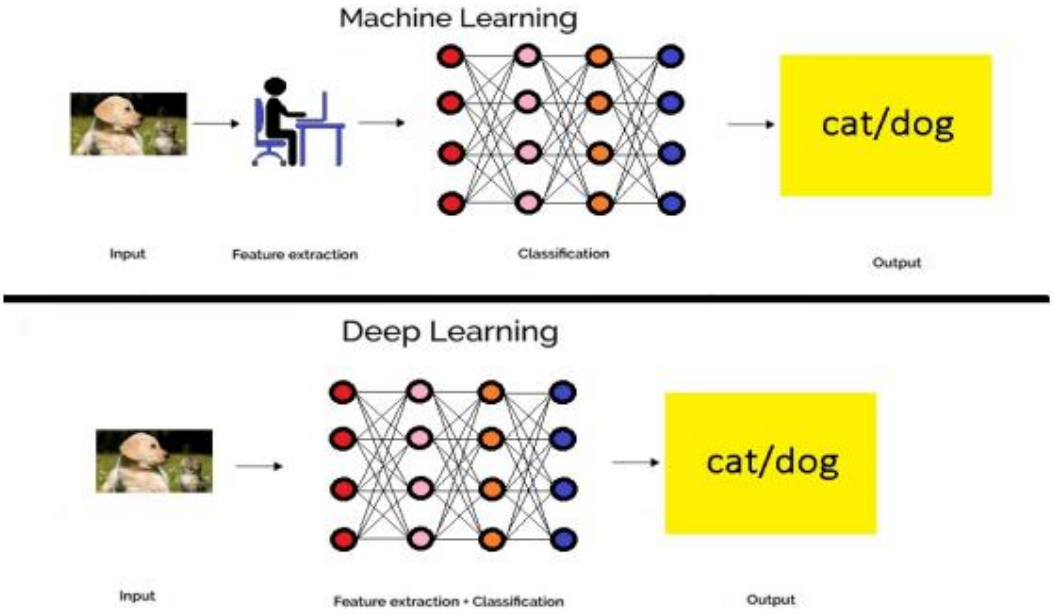
Derin öğreneme, Yapay Sinir Ağlarını kullanan makine öğrenimi metodolojilerinin belirli bir alt kümesidir ve Yapay Sinir Ağları insan beyninde bulunan nöronlardan esinlenerek oluşturulmuştur. YSA'nın katmanlarının çok fazla artırılması ile Derin Yapay Sinir Ağları oluştur [18].

Derin öğrenme algoritmalarının makine öğrenmesi algoritmalarına göre avantajı daha esnek olması ve ürüne dönüşümünde daha fazla kolaylık sağlamasıdır [19]. Derin öğreneme Avantajlarından birisi de veri seti eğitilirken eğitim tek katmanlı YSA kullanılması yerine gizli çok katmanlı YSA ulaşılan sonuç hata payını düşürür [20]. Derin öğrenme algoritmalarının makine öğrenmesi algoritmalarına göre dezavantajı ise daha güçlü donanıma gerek duymasıdır. Derin öğrenme algoritmaları çok fazla kombinasyonel işlem yapmakta ve bazen güçlü GPU(Grafik işlemci birimi) gibi donanımları gerekmektedir. [19].

Girdi olarak alınan resmin her pixeli girdi katmanın ayrı ayrı girdilerini oluşturur. Detay özellikler çıkarmak için bir dizi gizli katman oluşturulur. Girdi olarak alınan değer bir resim ise, pikseller göz önüne alındığında, ilk katman, komşu piksellerin parlaklığını karşılaştırarak kenarları kolayca belirleyebilir. İlk gizli katman kenarları bulduktan sonra, bu bulgulardan yola çıkarakikinci gizli katman, kenar koleksiyonları olarak tanınan köşeleri ve genişletilmiş könturları kolayca arayabilir. İkinci gizli katmanın resmin köşeler ve konturlar açısından açıklaması göz önüne alındığında, üçüncü gizli katman, belirli kontur ve köşe koleksiyonlarını bularak belirli nesnelerin tüm parçalarını algılayabilir. Son olarak, içerdiği nesne parçaları açısından görüntünün bu açıklaması, görüntüde bulunan nesneleri tanımak için kullanılabilir [21].

3.5. Derin Öğrenme ve Makine Öğrenmesi

Derin öğrenme makine öğrenmesinin alt dalıdır. Makine öğrenmesi derin öğrenmeye oranla daha geniş kapsamlıdır. Her iki yöntem arasındaki fark şekil 3.5'de gösterildiği gibi makine öğrenmesinde önce özellikler (feature) çıkarılıp sonra sistem sınıflama gibi bir işlem yaparken derin öğrenmede sistemin içinde yani derin öğrenmenin kendisi özellik çıkarımını yapar. Burada özellikte(feature) kasıt şudur ki; resimde verilen köpeğin kulağı, gözü, kuyruğu gibi köpeğe ait özelliklerdir.



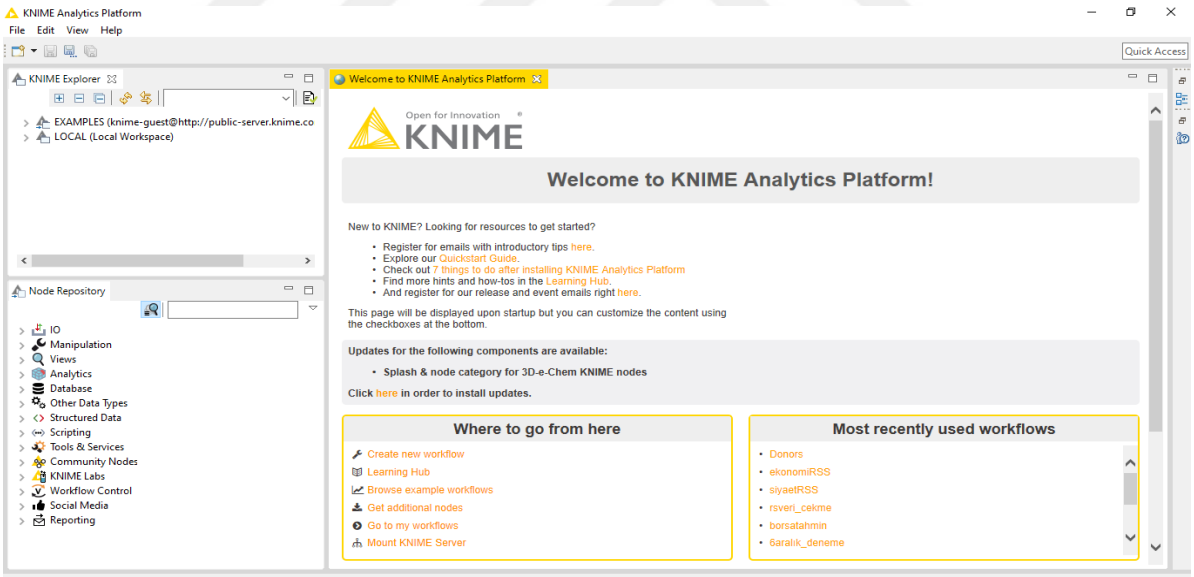
Şekil 3.5. Makine Öğrenmesi ve Derin Öğrenmesi [22]

4. KNIME

Genel olarak “Uçtan Uca Veri Bilimi” olarak bilinir. KNIME, veri bilimi sürecindeki her paydaşın en iyi yaptıkları şeye odaklanmasına olanak tanıyan kolay ve sezgisel bir ortam kullanarak veri bilimini oluşturmak ve üretmek için kullanılan bir yazılım platformudur [23]. Verinin görselleştirilmesi, makine ögrenmesi algoritması, birliktelik kurak çıkarımı gibi veri medenciliği araçlarının uygulanmasında kolaylık sağlayan açık kaynak kodlu veri bilimi yazılımıdır. Düğümlerden (node) oluşan bir çalışma akışından oluşur.

Knime’nin dünya çapında çok sayıda üyesi mevcuttur. Bu kullanıcılar yaptıkları çalışmaları Knime’in resmi sitesinde paylaşarak diğer kullanıcıların ulaşmasına olanak sağlarlar. Aynı zamanda yaptığınız herhangi bir çalışmada sorun yaşadığınızda platformun diğer üyelerinden yardım alabilirsiniz.

Knimen’nin kendi platform internet sitesinden indirilerek kurulum dosyası (setup) dosyasının çalıştırılması ile birkaç adımlıkkurulum aşamalarının sonunda Knime kullanıma hazır hale gelir. Şekil 4.1.’de Knimen’in açılış penceresi gösterilmiştir.

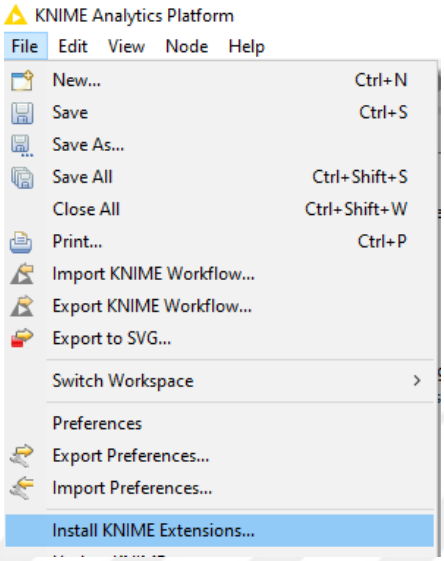


Şekil 4.1. Knime açılış bölümü

4.1. Install KNIME Extensions

“Install KNIME Extensions” Knime hakkında bilinmesi gereken en önemli bölümlerden biridir. Knime ilk kurulduğunda birçok düğümü(node) ve kütüphaneyi barındırmaz. Yapılan çalışmaya uygun güğümler(node) ve kütüphaneler kurulumdan sonra Knime’ye eklenmesi gerekir. Knime’ın iş akışlarını(workflow) oluştururken ihtiyaç duyduğumuz düğümleri

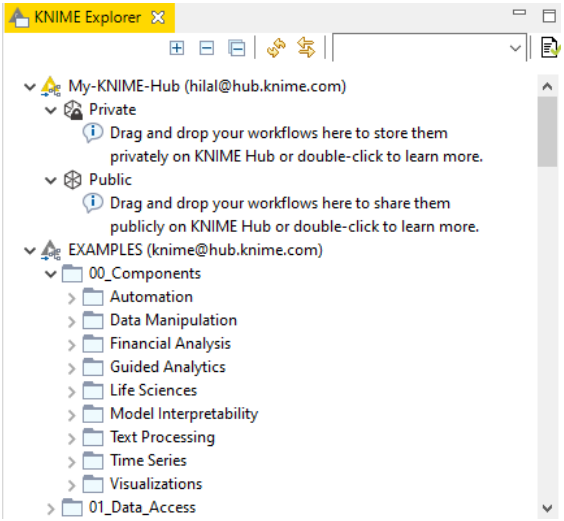
kullanabilmek bu düğümleri “Install KNIME Extensions” ile indirilmesi gerekir. Install Knime Extensions’na nasıl ulaşılacağı şekil 4.2.’de gösterilmiştir.



Şekil 4.2. Install KNIME Extension

4.2. Knime Hub ve İş Akışı(Workflow)

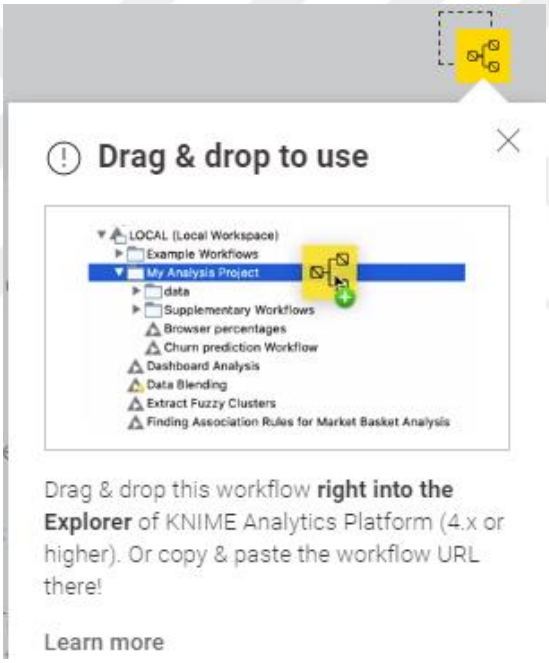
Knime’in Knime Explorer’ın alt sekmesinde barındırdığı “Knime Hub” bölümü büyük bir avantaj olan hazır iş akışı bölümüdür. Burada Knimeye kayıtlı dünya çapında çok sayıda veri bilimci veri bilimi için çözümler sunuyor. Yine kendilerinin yaptığı iş akışlarını(workflow) paylaşıyorlar. Şekil 4.3’de görüldüğü gibi çalışmalarınızı ister özel(private) isterseniz de herkese açık(public) yayınlayabilirsiniz.



Şekil 4.3. Knime Explorer

4.3. Drag & Drop To Use

Knime'nin sunduğu en büyük avantajlardan biri bırak ve kullan(Drag & Drop To Use) özelliğini kullanıcıya sunmasıdır. Açık kaynak kodlu bu platformun dünya çapında üyelerinin geliştirdiği iş akışları(workflow) bütün kullanıcıların hizmetine sunar. Knime'nin web sitesinde istediğiniz bir çalışma ile ilgili anahtar kelimeyi arama kısmına yazınca, bu anahtar kelimeyi barındıran çok sayıda çalışma size sunar. Bu çalışmaların hazır workflow'u yerel bilgisayara indirilebilir. Bunun için sürükle bırak ve kullan(Drag & Drop To Use) yöntemiyle istediğiniz iş akışını kolaylıkla kullanabilirsiniz. Knime web sitesine girdiğinizde yine Knime şekil 4.4'de gösterildiği gibi size bu sürükle bırak ve kullan(Drag & Drop To Use) işini nasıl yapacağınızı konusunda yardımcı oluyor. Bu kullanımla ilgili web sitesinde kullanıcıya ipucu veren küçük bir animasyon yer almaktadır.

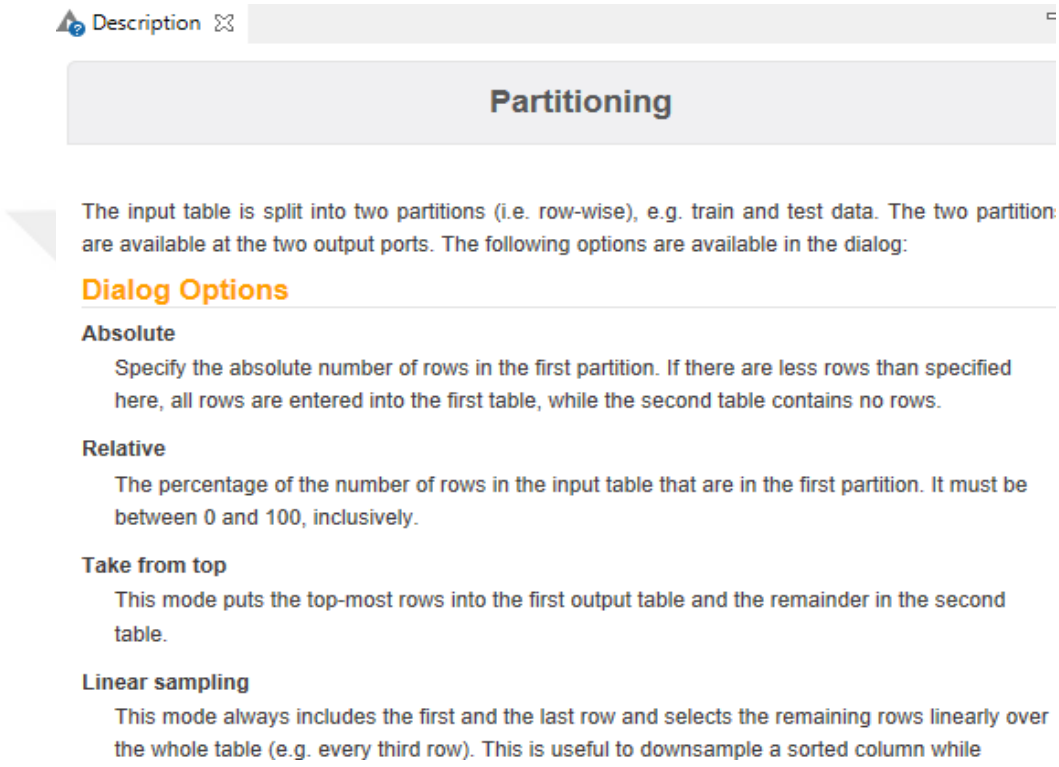


Şekil 4.4. Drag & Drop To Use

4.4. Knime Description Bölümü

Bu bölüm Knime'nin kullanıcılarına yardımcı olma bölümüdür. Kullanılan herhangi bir düğümün üzerine bir defa tıkladığınızda o düğüm için kullanıcıya açıklayıcı bilgiler sunar. Örneğin şekil 4.5'de "Partition" düğümü için; "Giriş tablosu iki bölüme ayrılmıştır (yani satır bazında), ör. verileri eğitin ve test edin. İki bölüm, iki çıkış bağlantı noktasında mevcuttur. İletişim kutusunda aşağıdaki seçenekler mevcuttur:İletişim Seçenekleri Mutlak İlk bölümdeki mutlak satır sayısını belirtin. Burada belirtilenden daha bağımsız satırlar varsa, tüm satırlar ilk tabloya girilirken, ikinci tablo satır içermez. İlk bölümdeki giriş tablosundaki satır sayısının

yüzdesi. 0 ile 100 arasında olmalıdır. Yukarıdan al Bu mod, en üstteki satırları ilk çıktı tablosuna ve kalanı ikinci tabloya koyar. Doğrusal örnekleme Bu mod her zaman ilk ve son satırı içerir ve kalan satırları tüm satırlar üzerinden doğrusal olarak seçer (ör. Her üçüncü satır). Bu, sıralı bir sütunu alt örneklemek için kullanışlıdır." Ve dahası olmak üzere çok bir veri bilimciye çok fazla bilgi sunmaktadır. Bu bölüm sayesinde kullanıcılar düğümlerin özelliklerine çok kolay erişebilmektedir.

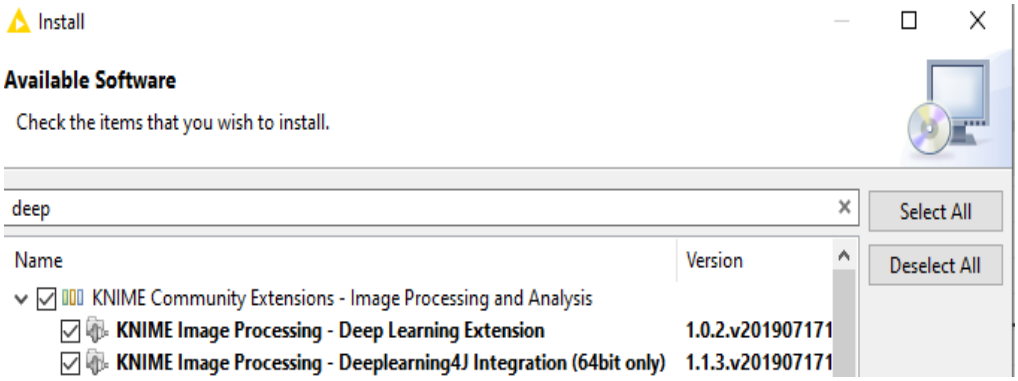


Şekil 4.5. Knime Descrition

4.5. Knime da Derin Öğrenme Kütüphaneleri

KNIME derin öğrenme uzantıları, KNIME Analytics Platformuna yeni derin öğrenme yetenekleri getiriyor. Bu derin öğrenme uzantıları, kullanıcıların KNIME Analytics Platformu içinde derin sinir ağlarını okumasına, oluşturmasına, düzenlemesine, eğitmesine ve yürütmesine olanak tanır [24].

Knime default olarak kurulduğunda Deep Learning kütüphaleri içinde barındırmaz. Knime'de Deep Learning uygulamalrı için öncelikle Install Knime Extension'dan gerekli kütüphaneler indirilmelidir. Bu tez çalışmasında kullanılan kütüphanelerin Knime'de seçilmesi şekil 4.6'da gösterilmiştir.



Şekil 4.6. Deep Learning kütüphanelerinin Knime'ye yüklenmesi

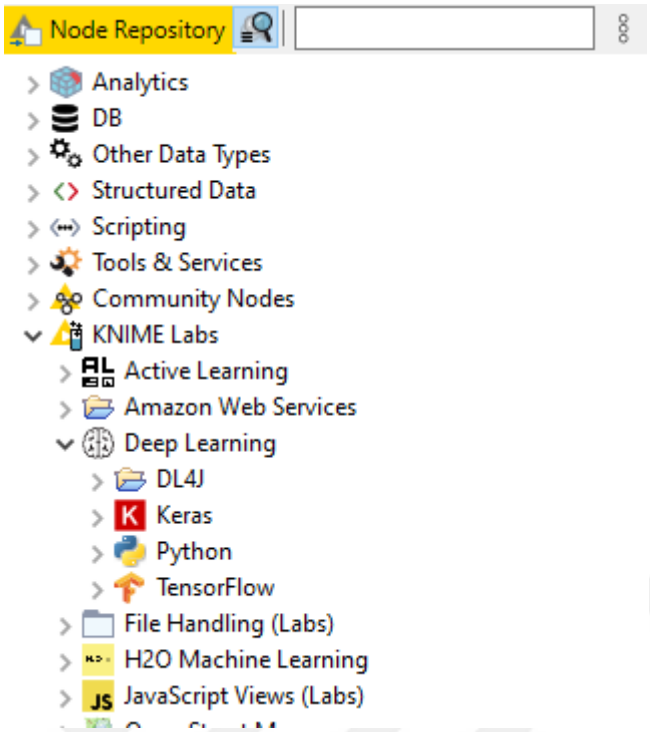
Knime'nin desteklediği kütüphanelerden "Deeplearner4J" kütüphanesi bu tez çalışmasında kullanılmıştır. "Deeplearner4J" kütüphanesi kütüphana java sanal makine(Java Virtual Machine-JVM) tabanlıdır.

Java sanal makinesi (Java virtual machine(JVM)), Java bayt kodunu çalıştırabilen sanal bir makinedir. JVM, Java yazılım platformunun kod yürütme bileşenidir [25].

DL4J deep learning kütüphanesi; en temel biçimde tanımlamak gerekirse Sinir Ağları'nı oluşturur, eğitir ve dağıtır ve bunu gerçekleştirirken için java programlama temeline dayanan bir araç setidir.. DL4J bir çok açık kaynak derin öğrenme kütüphanesinden yararlanan, hızlı bir şekilde çıktı geliştirmeye olanak veren açık kaynak kodlu bir kütüphanedir.

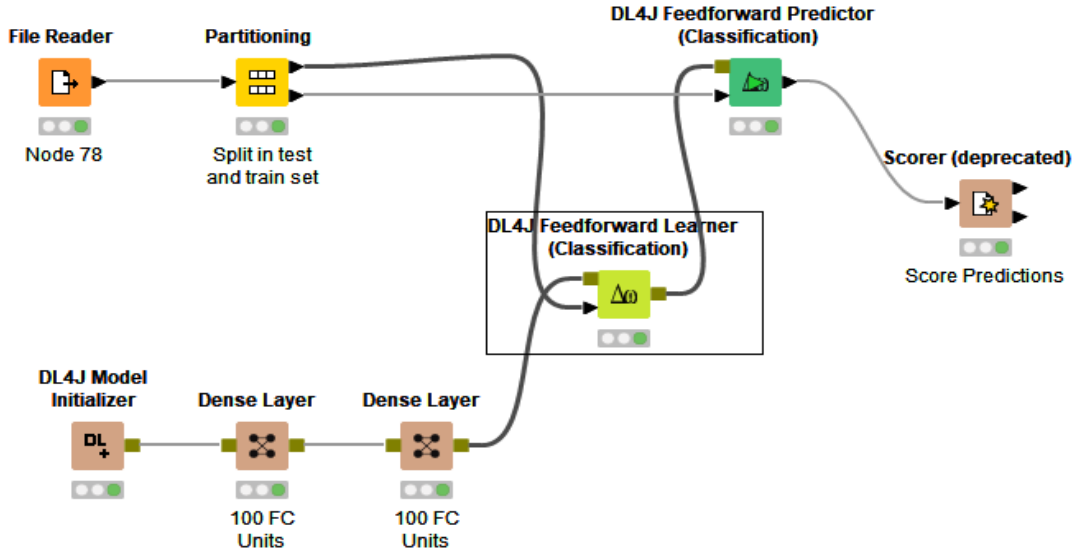
Knime Deeplearning4j Entegrasyonu, Deeplearning4j kütüphanesini Java'da derin öğrenme yetenekleri sağlayan KNIME ile bütünleşiktir. Knime'da, Deeplearning4j ağlarını okuyabileceğiniz, yazabileceğiniz, eğitebileceğiniz, çalıştırabileceğiniz ve inşa edebileceğiniz anlamına gelir [26].

"Install KNIME Extensions" ile düğümleri artık erişilmeye hazır olan Deeplearning4j kütüphanesi indirildikten sonra şekil 4.7'de gösterildiği "NodeRepository" bölümüne derin öğrenme düğümleri otomatik gelecektir. Deeplearning4j Entegrasyonu Node repository → Knime Labs → Deep Learning bölümünde bulunur. Aynı zamanda Node Repository arama çubuğu vasıtasıyla ihtiyaç duyulan düğüme kolay bir şekilde erişilebilir. Kolay kullanım olanağı sağlayan Knime'da düğümler sürükle bırak ile kullanıma hazır hale gelir.



Şekil 4. 7. Knime'de Node Repository

Knime Analistic Platform üzerinde Şekil 4.8'da Knime'de Deep Learning düğümlerinin bağlanması gösterilmiştir. Bu şekil 2 tane gizli katmanı olan bir derin öğrenme örneğidir. "File Reader" düğümü, çalışmada kullanılacak veri setinin sisteme yüklenmesini sağlar. Veriyi çeşitli formatları okuyacak şekilde yapılandırılabilir. "Partitioning" düğümü giriş tablosu iki bölüme ayrılmıştır (yani satır bazında), verileri eğitim ve test verisi gibi ayırır. bir giriş bölümü ve iki çıkış bağlantı noktasında mevcuttur. "DL4J Model Initializer" düğümü, bir ağ mimarisini başlatmak için kullanılan boş bir Derin Öğrenme Modeli oluşturur. Aynı zamanda düğüm her ağdaki ilk düğüm olması gerekir. Burada "Dense Layer" gizli katmanları temsil eder ve bu düğüm, giriş bağlantı noktası tarafından sağlanan Derin Öğrenme Modeline tam olarak bağlı bir katman ekler. "DL4J Feedforward Learner (Classification)" Bu düğüm, sınıflandırma için ileri beslemeli bir derin öğrenme modelinin denetimli eğitimini gerçekleştirir. "DL4J Feedforward Predictor (Classification)" düğümü; DL4J Feedforward Learner (Classification)'ın eğiterek oluşturduğu modeli ve test verilerini kullanarak bir sınıflandırma tahmini oluşturur. "Scorer" düğümü iki sütunu öznitelik değer çiftlerine göre karşılaştırır ve karmaşıklık matrisini gösterir, yani hangi öznitelik ve sınıflandırmalarının kaç satırının eşleştiğini gösterir. Aynı zamanda test verisinin doğruluk değerini gösterir."



Şekil 4. 8. Knime'de Deep Learning düğümleri

4.5.1. Knime de DL4J FeedForward Learner(Classification):

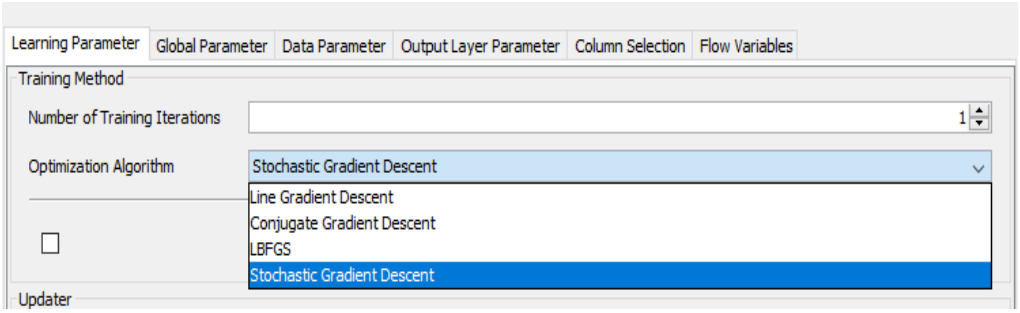
Bu düğüm, sınıflandırma için ileri beslemeli bir derin öğrenme modelinin denetimli eğitimini gerçekleştirir. Böylece, öğrenme prosedürü, düğüm iletişim kutusunda özelleştirilebilen çeşitli eğitim yöntemleri ve parametreleri kullanılarak ayarlanabilir. DL4J FeedForward Learner(Classification): düğümü, ağ yapılandırmasına otomatik olarak bir çıktı katmanı ekler ve bu katman, düğüm iletişim kutusunda da yapılandırılabilir. Sınıflandırma için, çıktı katmanı aktivasyon işlevi olarak her zaman 'softmax'ı kullanır ve çıktıların sayısı eğitim için kullanılan benzersiz etiketlerin sayısıyla eşleşecek şekilde otomatik olarak ayarlanır. Düğümün çıktısı, görünmeyen veri örneklerinin etiketlerini tahmin etmek için kullanılabilen eğitimli bir derin öğrenme modelidir [27]. Bu düğüm, beslendiği derin öğrenme ağı ve eğitim için ayrılan veri seti bölümünü kullanarak bir öğrenme gerçekleştirir. Bu düğümün oluşturduğu modeli "DL4J Feedforward Predictor (Classification)" düğümü kullanarak test veri setinden gelen verileri sınıflandırır.

4.5.2. Optimizasyon Algoritmaları:

Bir şeyi optimize etmek onu olabilecek en uygun duruma sokmak demektir. YSA'da en uygun değeri bulmak demek ağırlık(w) ve bias giriş parametreleri olan uygun değerler vermek demektir. Böylece buradaki asıl amaç olan maliyet(cost) azaltılmış olacaktır.

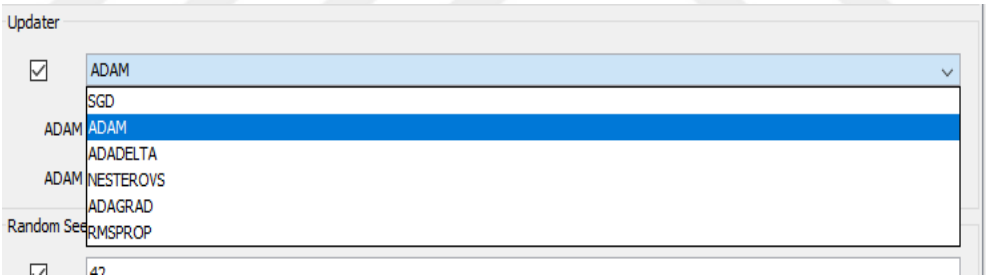
Bu tez çalışmasında kullanılan Schoastic Gradient Descent optimiztasyon algoritmasının seçimi şekil 4.9'da gösterilmiştir. Gradyan inişi, kayıpları en aza indirmek için genellikle ağırlık gibi öğrenilebilir parametreleri güncelleyen bir optimizasyon algoritması olarak kullanılır [10].

YSA eğitim veri seti üzerinde çalışırken verinin her bir satırı için ayrı ayrı çalışarak eğitir. Kullanılan girdi eğer bir resimse her resim için ayrı ayrı çalışarak bir sonuç bulur.



Şekil 4. 9. Knime'de Derin Öğrenme Sınıflandırma optimizasyon algoritması seçimi

Daha önce hesapladığımız maliyet fonksiyonu büyük çıkarsa sistem geriye salınım yaparak ağırlık(w), bias ve öğrenme katsayısı gibi parametreleri güncelleyerek yeni bir model oluşturur. Knime'nin sunduğu optimizasyon fonksiyonları şekil 4.10'da gösterildiği gibi stochastic gradient descent, adadelta, adam, adamax, adagrad ve rmsprop'dur.Optimizasyon algoritmaları seçimi Knime'de "DL4J FeedForward Predictor(Classification) " derin öğrenme düğümünden yapılır.



Şekil 4.10. Derin Öğrenmede Optimizasyon yöntemi seçimi

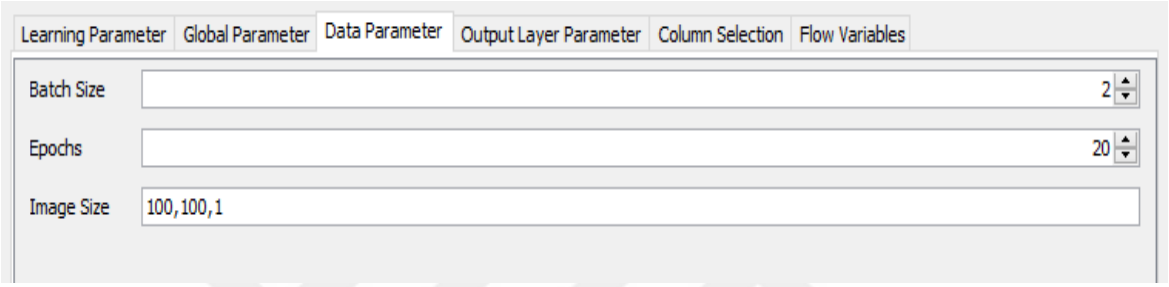
Knime da eğitim verisi olarak kullanılacak olan veri setinin veri parametrelerinin ayarlanması;

Batch(yığın/parça): kullanılan veri setini bir bütün olarak değil parça parça çalıştırır. Veri setini kaçar kaçar çalıştıracağımızı belirler. Örneğin kullanılan 40 resimden batch 2 seçilirse, optimizasyon algoritması çalışırken ileri ve geri salınımlar da her seferinde 2 resim üzerinde çalışır. Bu örnekte mevcut resim 40 ve her defasında 2 resimle üzerinde çalışıldığından 40/2=20 olur ve 20 devirde sistem kendini tamamlanır

Epoch(Devir): Epoch ise parçalara bölünen bu veri setinin toplamda kaç defa da biteceğini belirler. Epoch 3 verilirse 20 batch vardı böylece 20*3=60 batch olarak çalışır.

Epoch ve batch değerleri ne kadar büyük seçilirse doğruluk oranı artacağı gibi programın çalışma zamanı da artar.

İmage Size; Eğitim veri setine girdi olarak alınan resimlerin her biri farklı boyutlarda olabilir. Farklı boyutlar üzerinde çalışmayı kolaylaştırmak için resimler hepsi aynı boyutta olacak şekilde yeniden boyutlandırılır. Knime'da "Image Resizer" düğümü ile resimlerin yeni boyutunun ne olacağına karar verilir. "Data Parameter" seçiminde ise "Image Size" "Imege Resizer" değerleri ile aynı değerler seçilmelidir. Şekil 4.11'de epoch, batch ve image size parametrelerinin düzenlemesi gösterilmiştir.



Şekil 4.11. Veri Parametreleri ayarı

4.5.3. DL4J FeedForward Predictor(Classification):

DL4J Feedforward Predictor Classification) Bu düğüm, beslendiği derin öğrenme ağı ve test için ayrılan veri seti bölümünü kullanarak bir sınıflandırma tahmini oluşturur. Çıktı olarak, giriş tablosundaki her örnek için tahmin edilen sınıf değerini içeren bir sütun olarak karşımıza çıkar [28].

"DL4J Feedforward Predictor (Classification)" düğümü; DL4J Feedforward Learner (Classification)'ın eğiterek oluşturduğu modeli ve test verilerini kullanarak bir sınıflandırma tahmini oluşturur.

4.6. Karmaşıklık(Confusion) Matrisi

"Scorer" düğümü iki sütunu öznitelik değer çiftlerine göre karşılaştırır ve karmaşıklık matrisini gösterir, yani hangi öznitelik ve sınıflandırmalarının kaç satırının eşleştiğini gösterir. Aynı zamanda test verisinin doğruluk değerini gösterir.

Karmaşıklık(Confusion) Matrisi; test veri seti sonucu ile henüz test edilmemiş veri setinin karşılaştırılması sonucunu bir matris tablosunda sunar. Karmaşıklık(Confusion) Matrisi tablo test veri setinin sonucunu gerçek değer ve tahmin edilen yeni değerler ile karılaştırılmasını sunan bu tablonun şematiği tablo 4.1'de verilmiştir.

Tablo 4.1. Karmaşıklık(Confusion) Matrisi [29]

		Actual Values: Positive (1)	Actual Values: Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

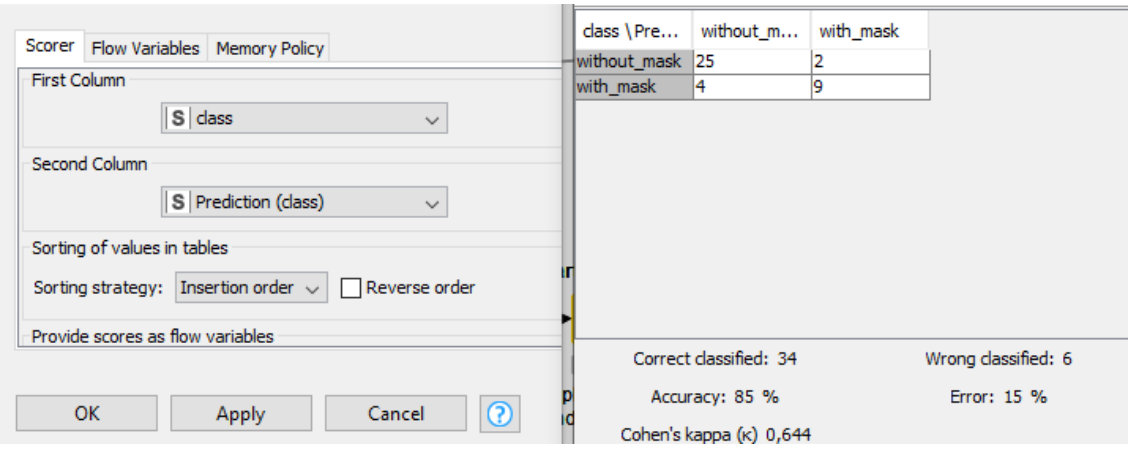
TP: Pozitif olan bir öğe pozitif olarak doğru tahmin etme

FP: Pozitif olan bir öğe pozitif değil olarak yanlış tahmin etme

FN: Pozitif olmayan öğe pozitif olarak yanlış tahmin etme

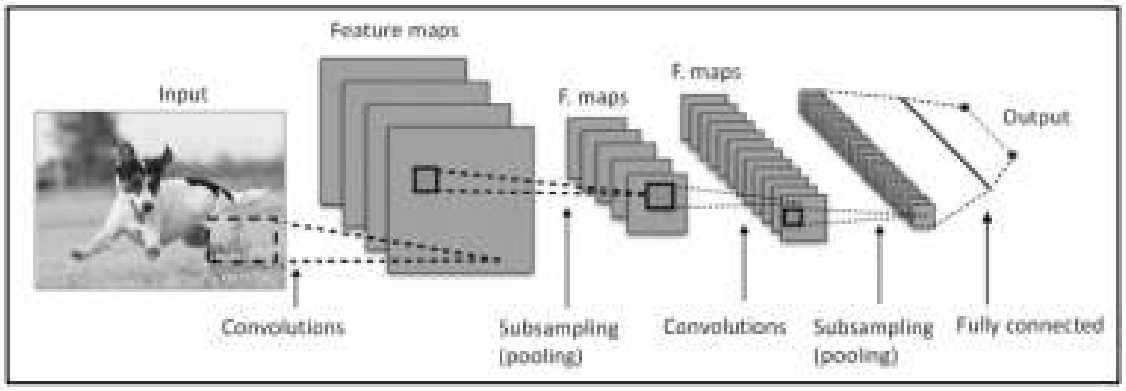
TN: Pozitif olmayan öğe pozitif değil olarak doğru tahmin etme

Knime'nin sunduğu "Scorer" düğümü Karmaşıklık(Confusion) Matrisini görmemizi sağlar. Bu düğümü sağ tıklayıp configure ettiğimizde çalıştığımız veri setinde hangi kolonu hangi kolonla kıyaslayacağımızı seçeceğiz. Şekil 4.12'de Knime'de yapılan uygulamanın Karmaşıklık(Confusion) Matrisi sonucu gösterilmiştir. Karmaşıklık(Confusion) Matrisi maskesiz kişilerin 25'ini doğru 2'ini yanlış tahmin etmiş ve maskeli kişilerin 4'ünü yanlış 9'ini doğru tahmin etmiştir.



Şekil 4.12. Knime'de Karmaşıklık(Confusion) Matrisi

5. DERİN ÖĞRENME

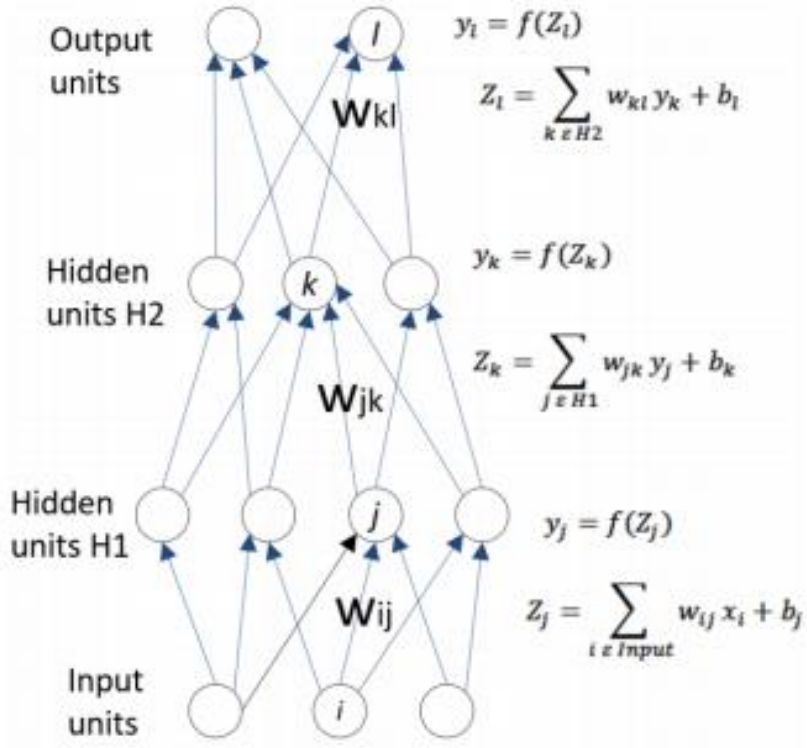


Şekil 5.1. CNN Mimarisi [30]

CNN(Convolutional Neural Network) Türkçe adıyla "Evrişimsel Sinir Ağları" temelde yapay zekâya dayanan doğruluk olasılığı yüksek derin öğrenme üzerinde çalışır. CNN daha çok resim tanıma ve bir görüntüden anlam çıkarma yöntemi olarak kullanılmaktadır. Resmin benzerleri üzerinde çalışarak daha fazla veriyi ilişkilendirmeyi sağlar. Örneğin Google da anahtar kelime arama yapıldığında görsellere tıklandığında aranılan resmin benzerlerini listelemesi veya daha önceleri arkadaşlarınızı Facebook resim etiketlenmesinde kişileri siz seçerken artık resmin arkadaşlarınız arasından kime ait olabileceğini size öneride bulunması gibi. Bir görüntüden anlam çıkarmada ise daha çok sağlık alanında kullanılmaktadır. Örneğin MR görüntülerinde tümör tespiti gibi. Şekil 5.1'de CNN mimarisinin siyah beyaz bir resimde iki boyutlu olarak gösterimi yapılmıştır.

Sinir Ağının Sınıflandırılması; İleri Beslemeli Sinir Ağı, Tekrarlayan Sinir Ağı (RNN), Radyal Temel Fonksiyon Sinir Ağı, Kohonen Kendi Kendini Düzenleyen Sinir Ağı, Modüler Sinir Ağı gibi farklı şekillerde sınıflandırılabilir [31]. Bu tez çalışmasında İleri Beslemeli Sinir Ağı (Feedforward Neural Network) üzerinde çalışılmıştır.

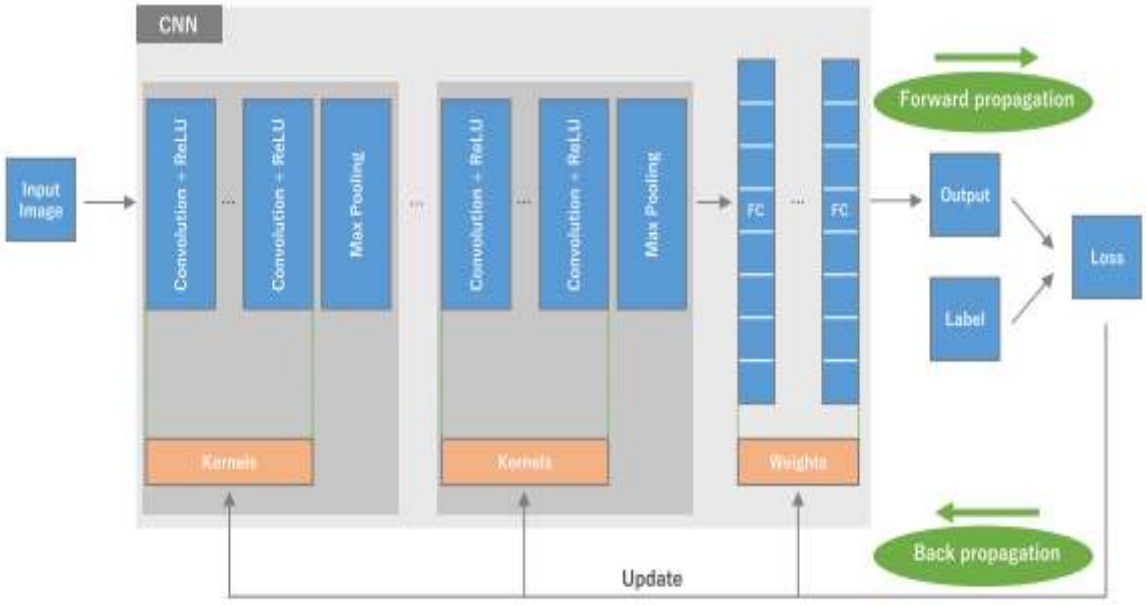
Şekil 5.2.'de, çok katmanlı bir uygulamanın belirli bir tipi olan ileri geçiş yolu boyunca hesaplanan değerler ve işlevlerle ileri beslemeli sinir ağı gösterilmektedir. Z, girdilerin ağırlıklı toplamıdır ve y, her katmanda Z'nin doğrusal olmayan aktivasyon fonksiyonunu(f) temsil eder. W, alt simge harfleriyle gösterilen bitişik katmanlardaki iki birim arasındaki ağırlıkları temsil eder ve b birimin önyargı değerini temsil eder [31].



Şekil 5.2. İleri Beslemeli Sinir Ağı [31]

5.1. CNN Adımları

Şekil 5.3' de görüldüğü gibi CNN mimarisi, evrişim katmanları, havuz katmanları ve tamamen bağlantılı katmanlar gibi birkaç bloktan oluşmaktadır.



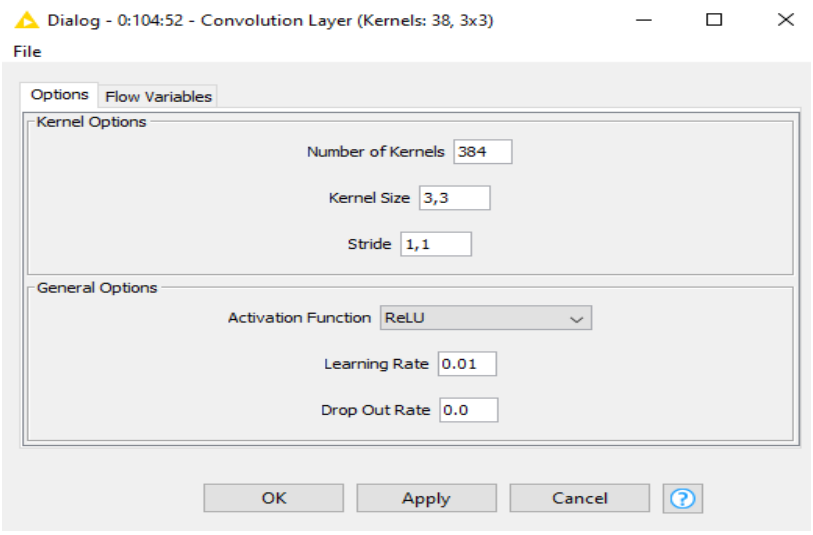
Şekil 5.3. CNN [10]

5.1.1. Birinci Adım: Konvolüsyon(Convolusion) İşlemi

Bir CNN ile ilgili olarak evrişim, geleneksel bir sinir ağına çok benzer şekilde, bir ağırlık koleksiyonunun girdiye çarpımından oluşan doğrusal bir işlemdir [32]. Görüntü CNN mimarisine bir matris olarak giriş yapar. Bu giriş matrisi bir filtre matrisi ile evrişim işlemine tabi tutulur. Her resim önce siyah beyaz yani iki boyutlu olarak sisteme girer ve eğer resim renkli ise resme üçüncü bir boyut olarak renk(rgb) boyutu eklenir. Resim işlemede renkli resimlerle çalışmak resme yeni bir boyut eklendiğinden resim 3 boyutlu bir matrise dönüşür. 3 boyutta çalışmak 2 boyutta çalışmaktan daha çok zaman aldığı için resimleri normalize ederek 3 boyuttan 2 boyuta dönüşüm sağlanır ve böylece yapılacak işlem çok daha kısa sürede yapılır. Burada yapılan işlemde resmin renk aralığı siyah-beyaz arasında yani gri tonlarda olur.

Çekirdek(kernel) çok katmanlı bir şekilde birbiri üzerine inşa edilmiştir bir dizi görüntü temsilini üretir [33]. Kullanılacak evrişim çekirdek sayısı yani katman tarafından üretilen özellik haritalarının sayısıdır. Şekil 4.3 görüldüğü gibi CNN sinir ağının kernel sayısını 384 olarak verilmiştir. Kernel sayısı ne kadar büyükse veri seti üzerinde CNN her kernel sayısı kadar çalıştırıldığından, programın çalışma süresi de o derece artar. Programın çalışma süresi aynı zamanda veri setinin büyüklüğü ile de doğru orantılı olarak değişir.

Adım(Stride), görüntülerin ve video verilerinin sıkıştırılması için ayarlanmış evrişimli sinir ağlarının veya sinir ağlarının bir bileşenidir. Adım, görüntü veya video üzerindeki hareket miktarını değiştiren sinir ağı filtresinin bir parametresidir. Mesela, bir sinir ağının adımı 1 olarak ayarlanmışsa, filtre bir seferde bir piksel veya birim hareket edecektir [34]. Şekil 5.4'te Adım sayısı 1 birim olarak seçilmiştir yani bir kernel her seferinde 1 birim ilerleyecektir. Burada kernel(filtre) boyutu 3x3 piksel olduğu için 9 piksellik çıktı katmanında 1 piksele dönüştürülür.



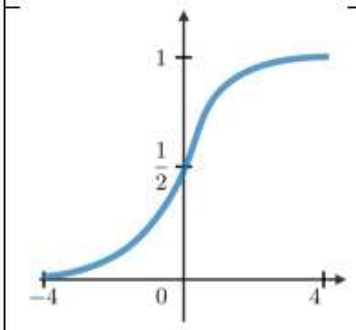
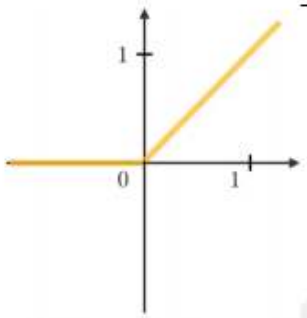
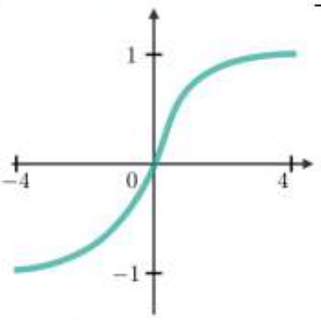
Şekil 5.4. CNN Kernel Sayısı

Düzeltilmiş doğrusal birim katmanı (ReLU), girdinin(g(z)) tüm elemanlarında kullanılan bir aktivasyon fonksiyonudur. Doğrusal olmamaları ile ağın öğrenmesi amaçlanmaktadır [35].

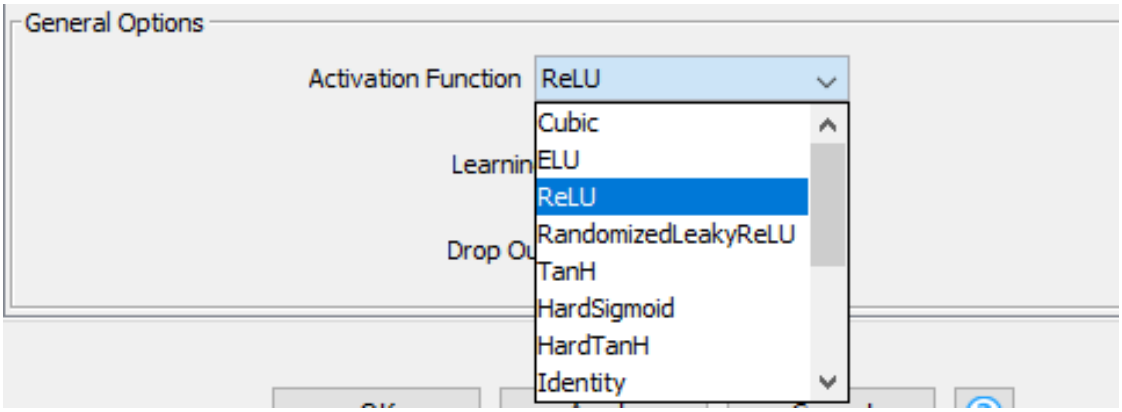
DL4J nin çeşitli aktivasyon fonksiyonları vardır. Bu aktivasyon fonksiyonlarından bazıları şunlardır; "ReLU, Sızıntı(Leaky), ELU: Exponential Linear Units Softsign, Softplus". Tablo 5.1'de "Sigmoid", "Relu" ve "Sofmax" aktivasyon fonksiyonlarının denklem ve grafikleri verilmiştir.

Bu tez çalışmasında aktivasyon fonksiyonu olarak RELU kullanılmıştır.

Tablo 5.1. Bazı Aktivasyon Fonksiyonu çeşitleri[36]

Sigmoid	RELU	Softmax
$f(x) = \frac{1}{1+e^{-x}}$ 	$f(x) = \begin{cases} x \text{ eğer } x \geq 0 \\ 0 \text{ eğer } x < 0 \end{cases}$ 	$f(x) = \frac{e_i^x}{\sum_i^k e_i^x}, i = 0,1,\ldots,k$ 

Bu uygulamada ReLU aktivasyon fonksiyonu kullanılmıştır. ReLU negatif değerlerle uğraşmamak için tüm negatif değerleri ortadan kaldırarak 0 alarak alır. Knime'de aktivasyon fonksiyonu seçimi Şekil 5.5'de gösterilmiştir.



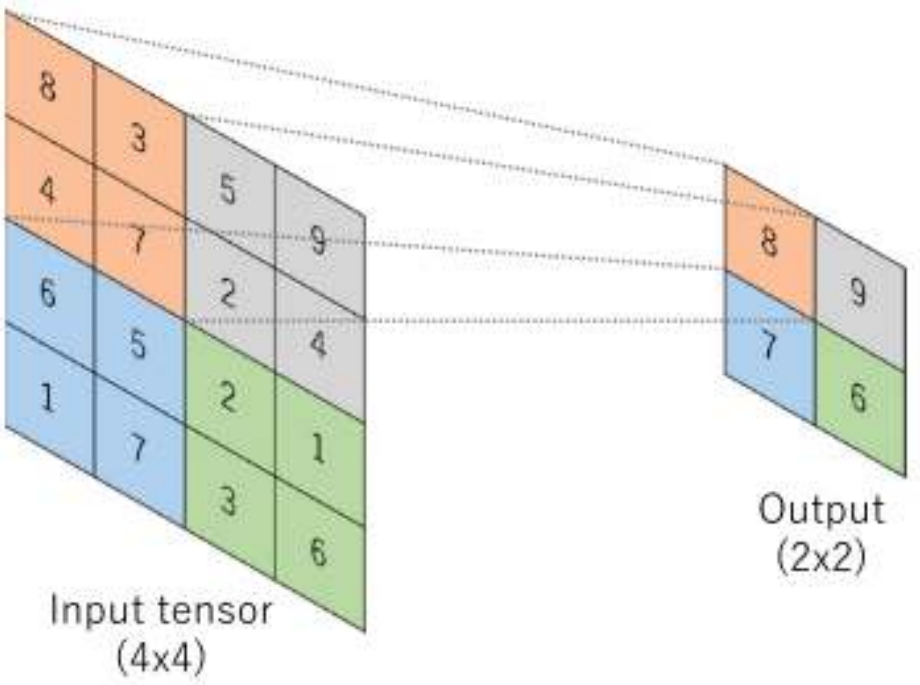
Şekil 5.5.CNN Aktivasyon Fonksiyon seçimi

5.1.2. İkinci Adım: Havuzlama(Pooling)

Görüntü işlemede havuzlama işlemi yapılırken önce kenar bulma, adım kaydırma ve son olarak havuzlama işlemi yapılır.

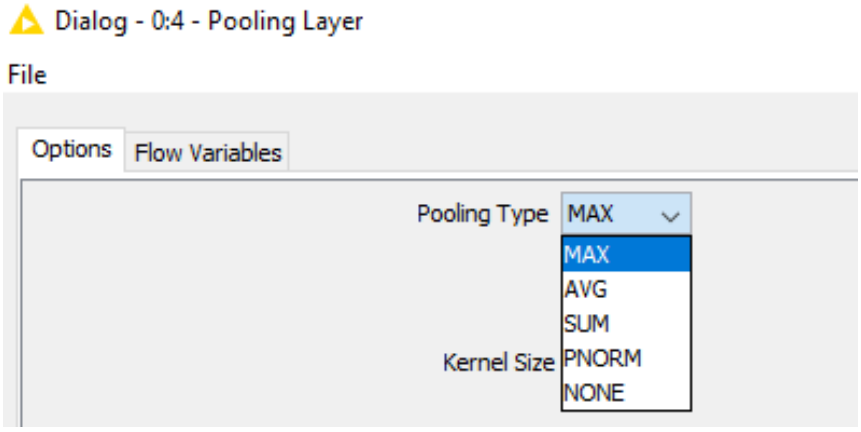
Kenar bulma işlemi, yoğunluk değişimlerinin olduğu bir yaklaşım kullanır. Yoğunlukta net ve tanımlanmış değişikliklerin meydana geldiği bir görüntüdeki noktaları tanımlamak için kullanılan bir dizi eylemdir [37].

Adım kaydırma işlemi; uygulanan filtrenin giriş matrisi üzerinde uygulama işleminde nasıl hareket edeceğini belirtir. Şekil 5.6'da giriş matrisi 4x4 luk 2 boyutlu bir matristir ve filtre matrisi ise 2x2 lik bir matristir. Giriş matrisi filtre matrisinden daha büyük bir boyuta sahip olduğundan burada matris çarpımı birden fazla yapılmalıdır. Adım sayısı 2x2 seçilmiştir. Filtre matrisi her defasında iki adım kaydırılarak uygulanmıştır. Yeni matris tipi olarak max pooling seçilmiştir. Yani her filtre işlemi uygulandığı adımdaki sonuç matrisinin en büyük olan değerine göre yeni bir matris oluşturulur. Oluşturulan bu yani matris giriş matrisden daha küçük bir boyuta sahip olur.



Şekil 5.6. Max Pooling Uygulaması[10]

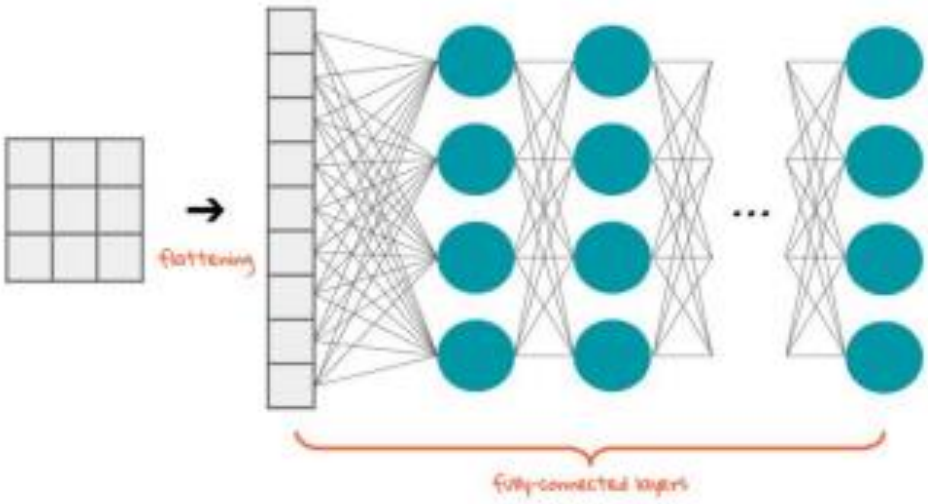
Havuzlama(pooling) işleminde seçilen pooling tipine göre yeni bir matris elde edilir. Şekil 5.7'de gösterildiği gibi Knime pooling tipleri gösterilmiştir.



Şekil 5.7. Pooling Layer

5.1.3. Üçüncü Adım: Flatting(Düzleştirme)

Düzleştirme, verileri sonraki katmana girilecek tek boyutlu bir diziye dönüştürmek için kullanılan bir işlemdir. Önceki katmanların çıktıyı, tek boyutlu bir vektör oluşturmak için düzleştirilir. Artık tüm pixeller tek bir satır olarak son katman verilir. Şekil.5.8'de CNN flatting gösterilmiştir.



Şekil 5.8. CNN de Flatting [32]

5.1.4. Dördüncü Adım: Fully Connected Layer(Tamamen Bağlı Katman)

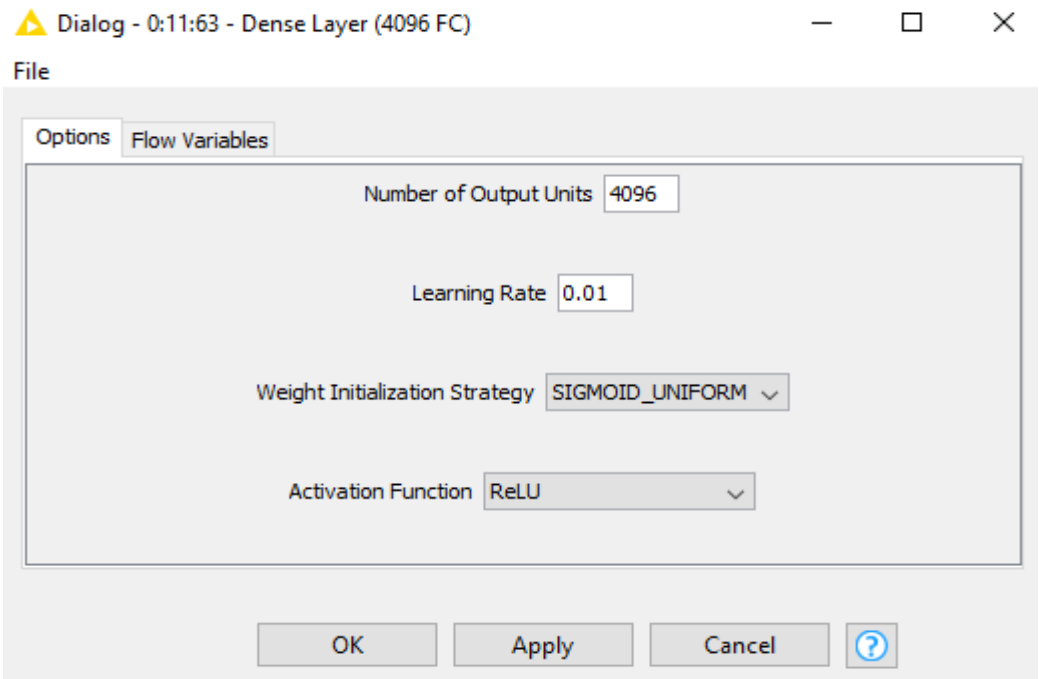
Bu katman, görüntüyü önceki katmanların sonuçlarıyla bir etikete sınıflandırmak için kullanılır. Evrişim ve havuzlamadan elde edilen değerler, her biri belirli bir özelliğin bir etiketin parçası olma olasılığını temsil eden bir değer vektörüne düzleştirilir [32]. Bu katmanın tam bağlı

katman olarak tanımlanması; önceki katmanda bulunan her nöronun bir sonraki katmanda bulunan her bir nörona bağlanmasından kaynaklanmaktadır [36]. Knime'de tam bağlı katman görevini "Dense Layer" düğümü gerçekleştirir.

5.2. Dense Layer

Yapay sinir ağının gizli katmanını temsil etmektedir. Birden fazla dense düğümü eklenerek gizli katman sayısı arttırılabilir. Her Dense Layer'lar birer gizli katmanı temsil etmektedir. Burada "Number of Output Units" sinir(nöron) sayısını temsil eder. Bu dense layer için 4096 tane nöron kullanılmıştır.

Öğrenme hızı Derin öğrenmede parametrelerin güncellenmesi geriye yayılım işlemi ile yapılmaktadır. Geriye yayılım işleminde yeni ağırlık değeri, geriye doğru türev alınarak hesaplanan fark değerinin öğrenme hızı parametresiyle çarpıldıktan sonra elde edilen değerin mevcut ağırlıktan çıkarılmasıyla hesaplanır. Öğrenme hızı sabit bir değer seçilebileceği gibi adım adım artan veya azalan şekilde de seçilebilir. Öğrenim hızının büyük seçilmesi modelin eğitim verisinden çok fazla etkilenmesine, küçük seçilmesi ise modelin daha yavaş ilerlemesine ve öğrenim süresinin artmasına neden olur [38]. Şekil 5.9'da Knime'de "Dense Layer" ayarlama(configurasyon) yapılması gösterilmiştir. Bu düğüm, giriş bağlantı noktası tarafından sağlanan Derin Öğrenme Modeline tam olarak bağlı bir katman ekler [39].



Şekil 5.9. Dense Layer

5.3. Eğitim ve Test Verisi

Veri madenciliğinde makine öğrenmesinin temeli verinin eğitilmesi esasına dayanır. Makine öğrenmesi daha öncede bahsettiğimiz gibi makineler insanlar gibi düşünmez, insanların öğrettiği daha doğrusu eğittiği gibi düşünen bir yapay zekâdır. Bu yapay zekâyı oluşturulurken veri setini eğitim ve test verisi olarak ikiye ayrılır. Veri seti ayrılırken dikkat edilmesi gereken en önemli nokta veri setini ayırma türüdür. Veriyi seti ikiye ayrılırken, veri setinin ilk yarısı ve ikinci yarısı olarak ikiye ayırma yapılması pek kullanışlı olmayabilir. Örneğin; şehirlerde suç olması durumunun incelendiği bir projede şehirler alfabetik olarak sıralanmış ise veriyi üstten başlayarak ikiye bölme işlemi yapmak doğruluk oranımızı düşürme olasılığı yükselir. Burada en çok tercih edilen ayırma türü randomly (rastgele) ayırma türüdür.

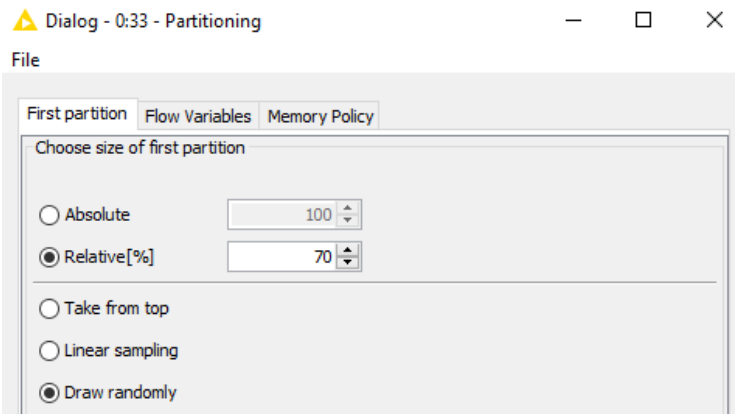
5.3.1. Eğitim Verisi(Training Data):

Eldeki verilerden yola çıkarak belli sonuçlara ulaşılması veri setinin eğitilmesidir. İşte buradaki kullanılan veri setine “Eğitim veri seti” denir. Eğitim verisi veri setinin tamamı olarak seçilebileceği gibi veri setinin belli bir bölümünden de seçilebilir.

5.3.2. Test Verisi(Test Data)

Veri setimizin ikinci bölümü; test veri seti bölümdür. Yapay zekâmızın eğitim veri seti ile karşılaştırma yaparak bir çıkarımda bulunmasını sağlayan veri setine “Test veri seti” denir.

Knime’de veri setimizi eğitim ve test veri seti olarak ikiye ayırmayı sağlayan “Partitioning” ve “X Partitioning” düğümleri vardır. Biz bu çalışmamızda randomly ayırma türü olanağı sağlayan “Partitioning” düğümünü kullanacağız. Şekil 5.10’da Knime’de veri setinin %70’lik kısmı “Eğitim veri seti” ve %30’luk kısmı test veri olarak randomly ayrılması gösterilmiştir. Eğitim ve test veri setleri aynı dosyadan gelebileceği gibi farklı dosyalardan da gelebilir.

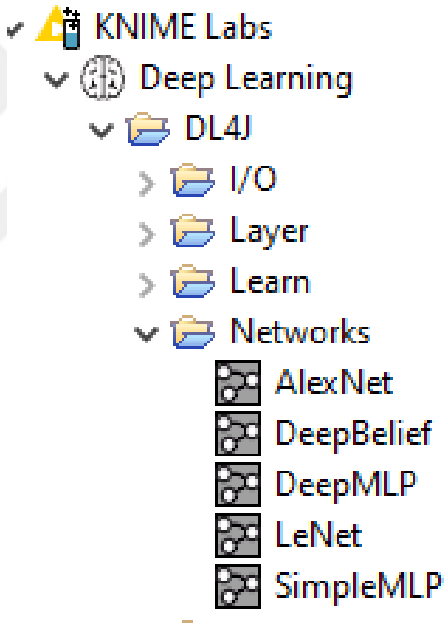


Şekil 5.10. Veri Setini Eğitim ve Test verisi olarak ayırma

"Pratitioning" düğümü yardımı ile Eğitim ve Test verisi olarak ikiye ayırdığımız veri setimizin ilk bölümü yani eğitim veri setini kullandığımız "DLJ4 Feedforward Learner (Classification)" düğümüne girdi olarak verilir. "Pratitioning" düğümünün ikinci kısmı test veri setini kullandığımız "DLJ4 Feedforward Predictor (Classification)" düğümüne girdi olarak verilir.

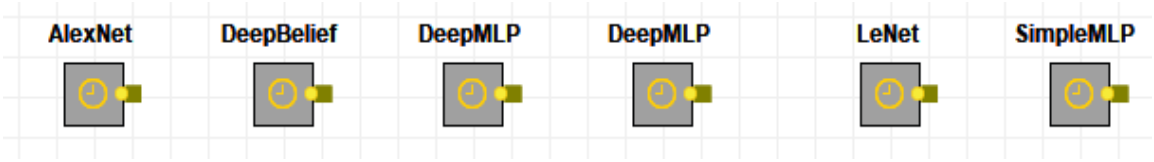
5.4. Derin Öğrenme Mimarileri

Yukarıda bir convolution neural network içerisinde yapılan işlemler adım adım anlatıldı. Sıra derin öğrenme mimarisini oluşturmaya geldi. Knime bu mimarileri hazır olarak sunuyor. Knime'da derin öğrenme mimarilerine "Node Repository" kısmından ulaşılabilir. Şekil 5.11'de Knime'nin sunduğu mimariler gösterilmiştir.



Şekil 5. 11. Knim'da Derin Öğrenme Mimarileri

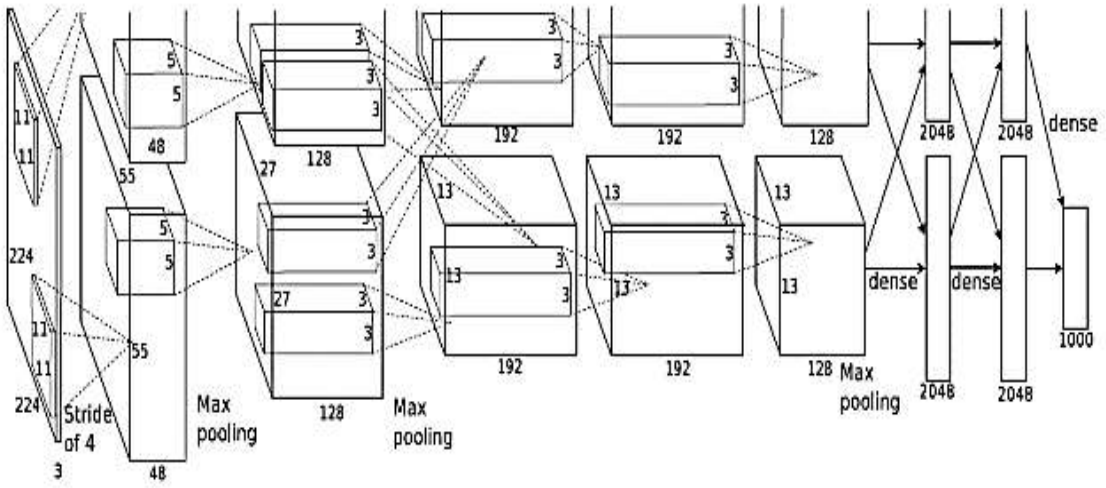
Knime sürükle bırak kolay kullanımı sayesinde istenen herhangi bir düğüm çalışma ekranına taşınarak kullanılabilir. Knime'nin hazır olarak sunduğu derin öğrenme mimarilerinin çalışma ekranına taşınmış halleri şekil 5.12'de verilmiştir. Knime platformu çalışmaların daha derli toplu yapılmasını sağlar. Örneğin tek bir ekranda çalışmanın her düğümünün bulunması anlam karmaşasına sebebiyet vereceğinden düğümler kendi alt dallarında faklı sayfalarda saklanır. Böylece hem çalışmayı yorumlamak hem de projede çalışma daha kolay olacaktır.



Şekil 5.12. Knime'de Derin Öğrenme Mimarileri Düğümleri

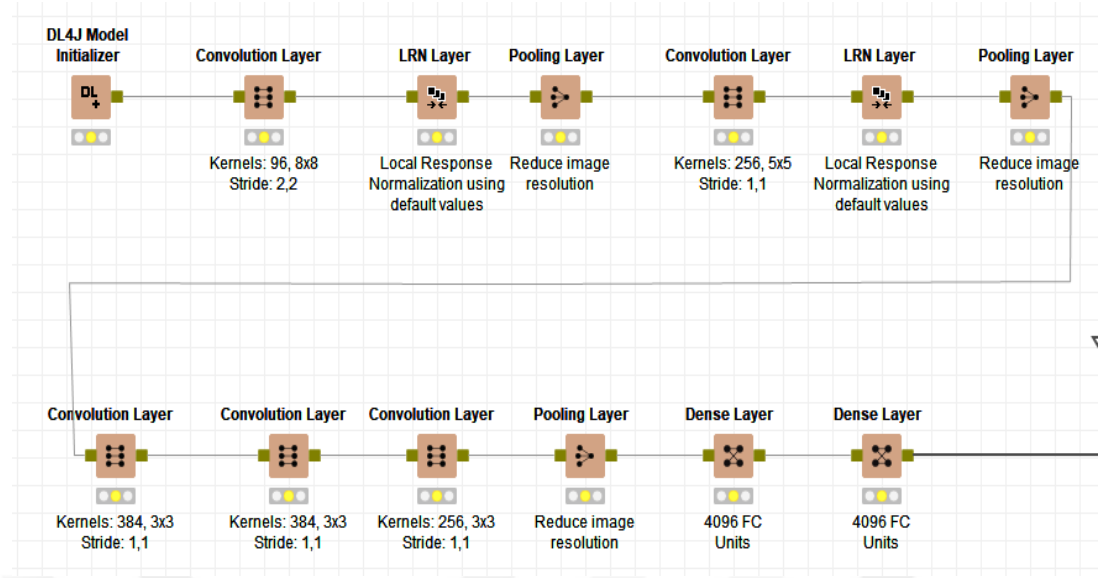
5.4.1. AlexNet

Birbirini takip eden evrişim ve ortaklama katmanları bulunmasından dolayı, temel olarak LeNet derin öğrenme modeline benzemektedir ve Lenet 'in gelişmiş şeklidir. Aktivasyon fonksiyonu olarak ReLU (Rectified Linear Unit), havuzlama(Pooling) tip olarak max pooling tipi kullanılmaktadır [13]. Şekil 5.13'de AlexNet mimari yapısı gösterilmiştir.



Şekil 5.13. AlexNet ağı [36]

Knimede düğüm deposunda(node Repository) hazır olarak sunulan "AlexNet" düğümünün(node) açık şekil 5.14'de verilmiştir. Bu şekil yukarıda verilen şekil 5.13'de AlexNet mimari yapısının Knime' ye uyarlanmış halidir. Knime'de "DL4J Model Initializer" düğümü ile derin öğrenme başlatılır. Bu düğümün devamında AlexNet sırasıyla "Convolution Layer", "LRN Layer", "Pooling Layer" gibi düğümlerden oluşur. Her düğümün konfigürasyonu Knime tarafından otomatik yapılmıştır. Gerektiği durumlarda düğümlerin konfigürasyon ayarları değiştirilebilir.

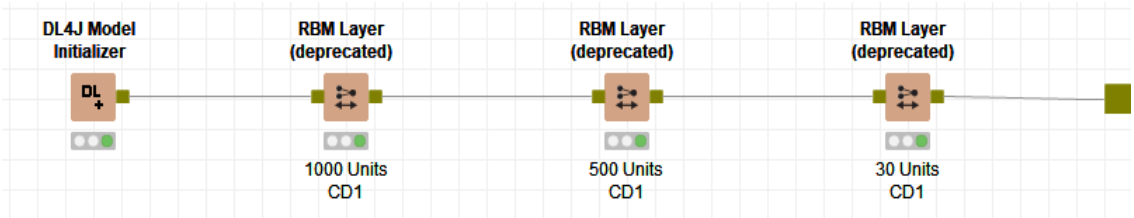


Şekil 5.14. Knime'de AlexNet iç yapısı

5.4.2. Derin İnanç Ağları(Deep Belief Network)

Restricted Boltzmann Machine(RBM) de gizli katmanlar arasında bağlantı yoktur. Birden fazla gizli katman, bir RBM'nin gizli aktivitelerini daha yüksek seviyeli bir RBM'yi eğitmek için veri olarak ele alarak eğitebilir [40].

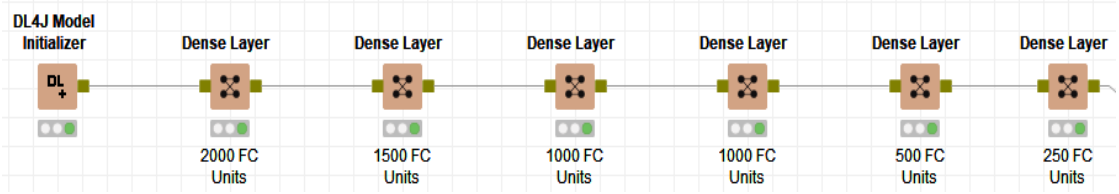
Knime Derin inanç ağını "DeepBelief" düğüm altında paket olarak sunuyor. Node Repository"den bu düğüme ulaşmak mümkündür. Bu düğüme çift tıklandığında karşımıza şekil 5.15'de çıkmaktadır



Şekil 5.15. Knime'de Deep Belief iç yapısı

5.4.3. DeepMLP(Derin Çok Katmanlı Algılayıcı (Deep Multi Layer Perceptron))

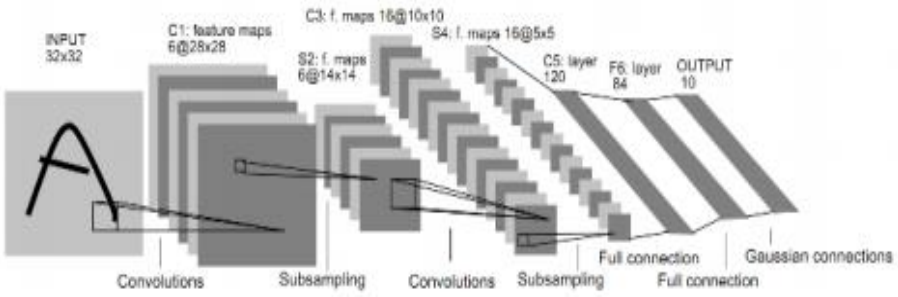
Knime'de "DL4J Model Initializer" düğümü ile derin öğrenme başlatılır. Bu düğümün devamında 6 tane "Dense Layer" yani gizli katman bağlanması ile oluşur. Mlp ileri beslemeli bir sinir ağıdır. Şekil 5.16'da Knime'de Depp MLP düğümlerinin hazır olarak birbirine bağlanması gösterilmiştir.



Şekil 5.16. Knime'de DeepMLP iç yapısı

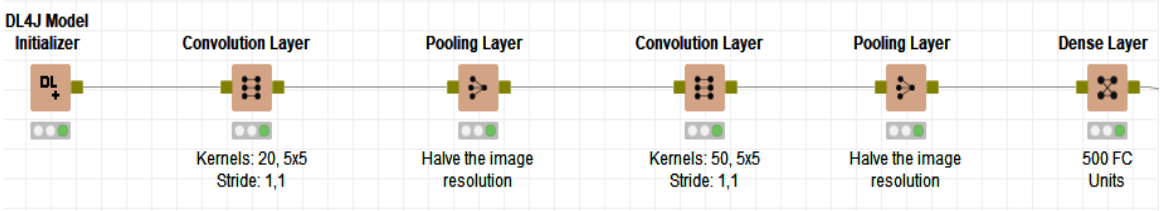
5.4.4. LeNet-5

LeNet-5, gradyan tabanlı bir öğrenme CNN yapısıdır ve ilk olarak elle yazılmış dijital karakter tanımada başarıyla uygulanmıştır. daha çok 80'li yıllarda imza ve el yazısını tanımlayabildiğinden bankalar tarafından çek senet doğrulamasında kullanılmıştır [20]. Şekil 5.17'de LeNet mimarisinin yapısı görülmektedir. Burada bir dijit verisinin sınıflandırılması için bir mimari örneği gösterilmektedir [41]. Şekilde de görüldüğü gibi havuzlama(pooling) tipi olarak ortalama(avg) pooling tipini kullanır.



Şekil 5.17. Lenet [41]

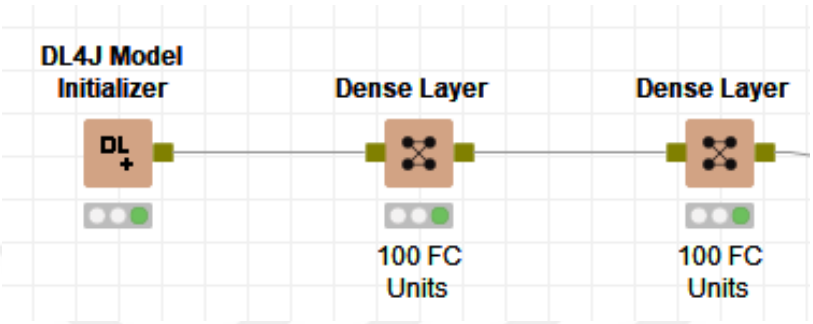
Şekil 5.18' de Knime uygulama programında "LeNet" düğümünün hazır iç yapısı gösterilmiştir. Knime'de "DL4J Model Initializer" düğümü ile derin öğrenme başlatılır. Bu düğümün devamında Lenet sırasıyla "Convolution Layer", "Pooling Layer" "Convolution Layer", "Pooling Layer" ve "Dense Layer" düğümlerinden oluşur.



Şekil 5.18. Knimede LeNet Düğümü

5.4.5. SimpleMLP (Basit Çok Katmanlı Algılayıcı (Simple Multi Layer Perceptron)

Sinir ağlarının ilkel halı gibi düşünülebilir. Knime'nin hazır sunduğu "SimpleMLP" 5.19'da iki alt katmandan oluşan bir derin öğrenme mimarisi oluşturmaktadır. Knime'de "DL4J Model Initializer" düğümü ile derin öğrenme başlatılır. Bu düğümün devamında SimpleMLP 2 tane gizli katman yani "Dense Layer" düğümünün bağlanması ile oluşur.



Şekil 5.19. Knime'de SimpleMLP düğümünün iç yapısı

Knime'nin hazır olarak sunduğu derin öğrenme mimarilerinin yanı sıra istenen düğümler seçilerek gerekli konfigürasyonlar yapılarak Knime'de derin öğrenme çalışmaları yapılabilir.

6. KOLLEKTİF ÖĞRENME(ENSEMBLE LEARNİNG)

Veriden anlamlı ve işe yarar bilgiyi elde etmek için hali hazırda birçok yöntem vardır. Bu yöntemlerin başında Makine öğrenmesi algoritmaları ve derin öğrenme yöntemleri gelir. Fakat aynı algoritma farklı veri setleri üzerinde çalıştırıldığında farklı sonuçlar vermektedir. "Bütün kendini oluşturan parçalardan büyüktür". Felsefesinden yola çıkarak bu çalışmada bir çok algoritmadan aynı anda toplu olarak yararlanmaya çalışılmıştır. Böylece topluluğun gücüne dayanan kolektif öğrenme eğitim ve test verileri üzerinde çalışan birden fazla farklı algoritmanın en verimli olanın seçilmesini sağlar. Hem zamandan ki son yıllarda zamandan kazanç çok daha önem kazanmıştır, test verileri doğruya daha yakın olma olasılığını arttırıyoruz.

Bir kolektifin(ensemble) kendisi denetimli bir öğrenme algoritmasıdır, çünkü bu kolektifin kendisi eğitilebilir ve daha sonra tahminlerde bulunmak için kullanılabilir. Eğitimli topluluk, bu nedenle, tek bir hipotez. Ancak bu hipotez, kendini oluşturan modellerin hipotez uzayında oluşmaktadır. Böylece topluluklar temsil edebilecekleri işlevlerde daha fazla esnekliğe sahip oldukları gösterilmelidir. Bu esneklik, teorik olarak, eğitim verilerini aşırı uydurmalarına, daha fazla uymalarına olanak sağlayabilir. Tek bir modelden daha fazla modelin çalıştırılan topluluk teknikleri, toplulukta en iyi sonuçlar verme eğilimindedir [42].

6.1. Deeep Ensemble Learning

Derin CNN kollaktif(ensemble)de alt küme oluşturulurken, orijinal alt kümenin rastgele seçilmiş verileri arasından seçilir ve herbiri ayrı bir CNN sınıflandırıcı tarafından olşturulur daha sonraki aşamada yine herbiri kendine kaşılık gelen alt küme içinde hem eğitilir hem de test edilir [43].

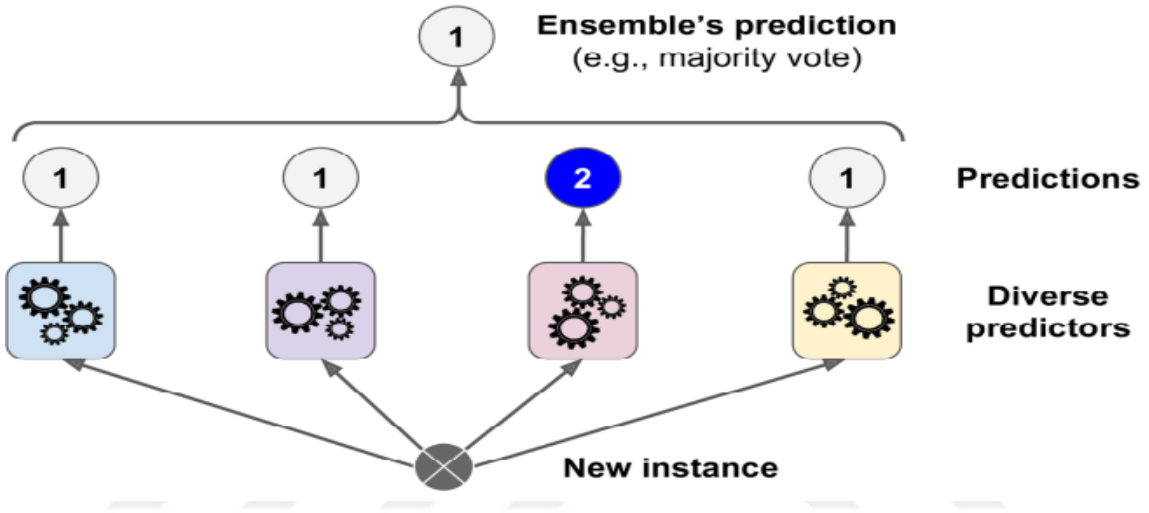
6.2. Ensemble Learner Teknikleri

Ensemble learner teknikler birden fazla yöntemi kullanarak bu yöntemlerden doğruluk derecesi en yüksek olanın yada kullanılan yöntemlerin ortalaması alınarak sonuç çıktısına karar verme gibi yöntemlerdir.

6.2.1. Çoğunluk Oylaması(Majority Voting(MAVL))

Çoğunluğun dediği olur." oylama yöntemi mantığı üzerine çalışır. Birden fazla algoritma bir eğitim seti üzerinde çalışır ve yeni gelen örnek sisteme giriş yaptığında sistem bu yeni gelen test örneği için her algoritmayı çalıştırır. Bu algoritmalar yeni gelen test örneğinin hangi sınıfa ait olduğunu ayrı ayrı hesaplar. Bu hesaplamalar sonucunda hangi sınıf daha çok oy almış ise yeni

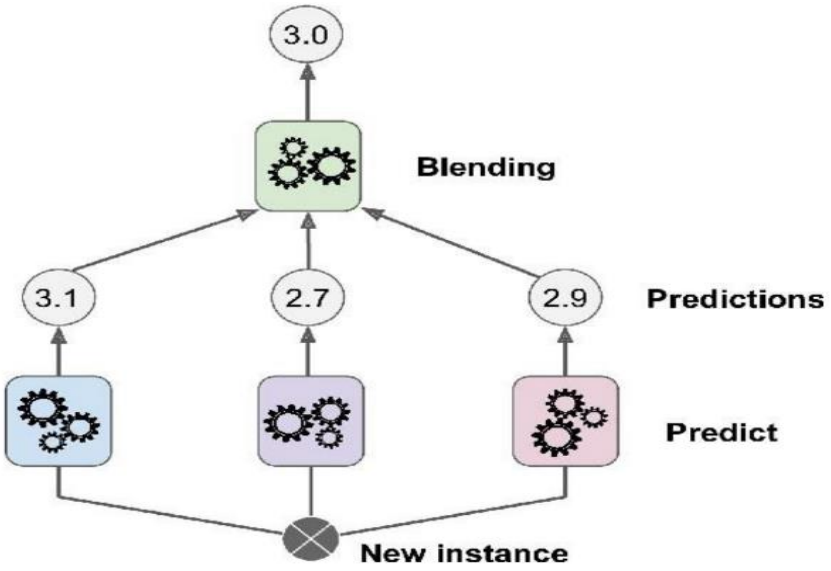
test verisi o sınıfa dahil edilir. Şekil 6.1'de dört farklı algoritma arasında çoğunluk oylaması yapılmış ve test sonucu 3 farklı algoritmaya göre 1. sınıfı, 1 algoritmaya göre 2. sınıfta tahmin edilmiştir. MAVL'ye göre 3'e 1 çoğunluğundan test verisi 1. Sınıfta olduğu kabul edilir. MAVL'de algoritmalara ağırlık ataması yapılarak da çalıştırılabilir. MAVL'de algoritmaların yüzdelik olarak büyük olanı da alınabilir. Örneğin şekil 6.1'de 1. ve 2. algoritmalar %50 3. Algoritma %60 ve sonuncu algoritma %90 oranında tahminde bulunursa bu tahminlerden en büyük olanı %90 olduğundan test örneği 4. sınıfa dahil edilir.



Şekil 6.1. Majority Voting [44]

6.2.2. Stacking (Yığınlama) Yöntemi

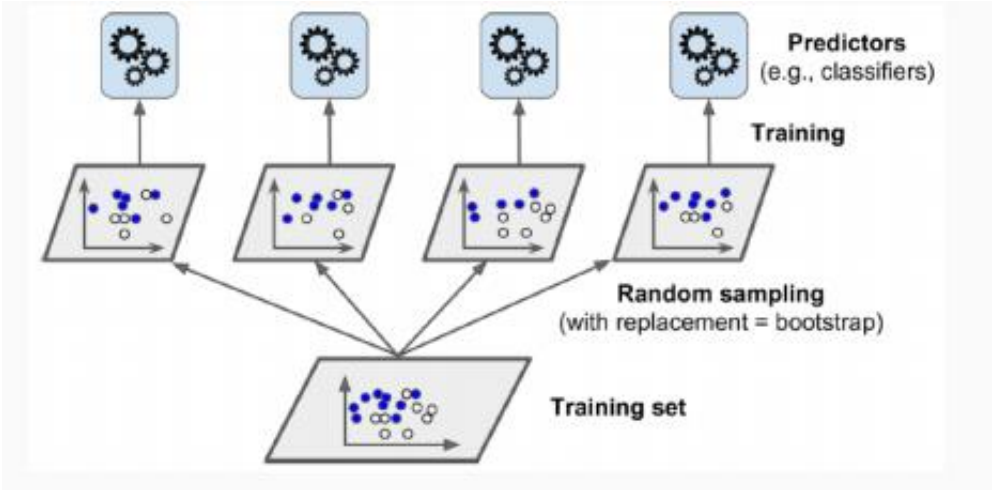
Stacking (yığınlama) yöntemi klasik bir ensemble learner yöntemidir. Bu yöntemde istenirse çalıştırılacak algoritmalara ağırlık ataması yapılabilir. Şöyle ki; bir numaralı algoritmanın çalışmasına daha güvenilir ise diğerlerine oranla, bir numaralı algoritmaya 3 ağırlık, iki numaralı algoritmaya ise güvenilirlik açısından diğerlerine oranla bir miktar daha düşük olduğundan 1 gibi ağırlık ataması yapılabilir. Şekil 6.2'de birinci algoritmaya 3.1, ikinci algoritmaya 2.7 ve üçüncü algoritmaya 2.9 ağırlık ataması yapılmıştır. Stacking sayısal tahminler için kullanılmıştır. Burada tahmin sonuçlarının ortalama noktası alınır. Böylece sisteme yeni giren test verisinin hangi sınıfa dahil olacağına karar verilir.



Şekil 6.2. Stacking [45]

6.2.3. Bagging (BootStrap Aggregation) Yöntemi

Bagging yöteminde algoritmalar eğitilirken veri bütün halinde girdi olarak verilmez. Algoritma örnekleri üzerinde veri değiştirilerek eğitim işlemi yapılır. Veriyi değiştirmek için, veriyi bölme ya da verinin bir kısmını değiştirme gibi yöntemler uygulanabilir. Her algoritma kendi veri çalışma bölgesi için tahminde bulunur. Burada her bir algoritmanın bir ağırlığı vardır. Algoritmaların bulduğu sonuçlar bu ağırlıklarla çarpılarak çalışır. Şekil 6.3'de Bagging (BootStrap Aggregation) yöntemi gösterilmiştir.

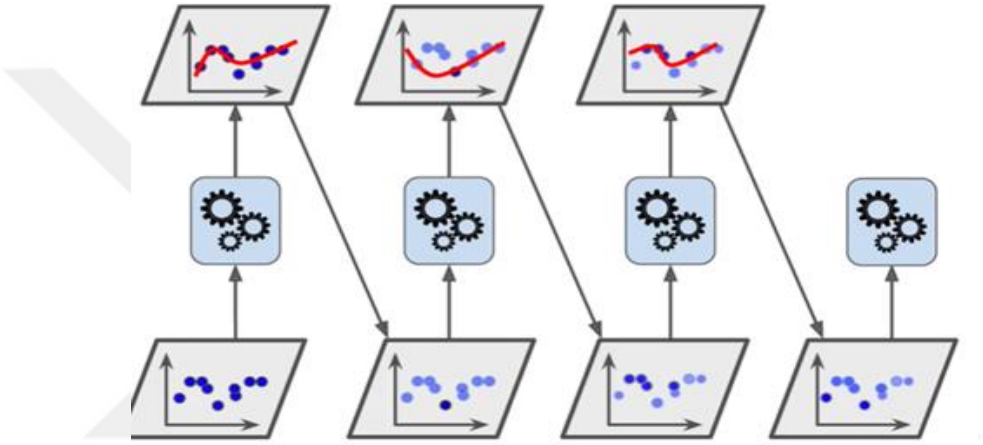


Şekil 6.3. Bagging [46]

6.2.4. AdaBoost Yöntemi

Sisteme giren algoritmalardan güvenilirliği daha yüksek olana diğerlerine oranla daha büyük bir ağırlık algoritması verilerek çalışır. Daha sonra sonuçlar gözlemlenir ve daha iyi sonuç veren algoritmanın ağırlığı pozitif yönde güncellenir.

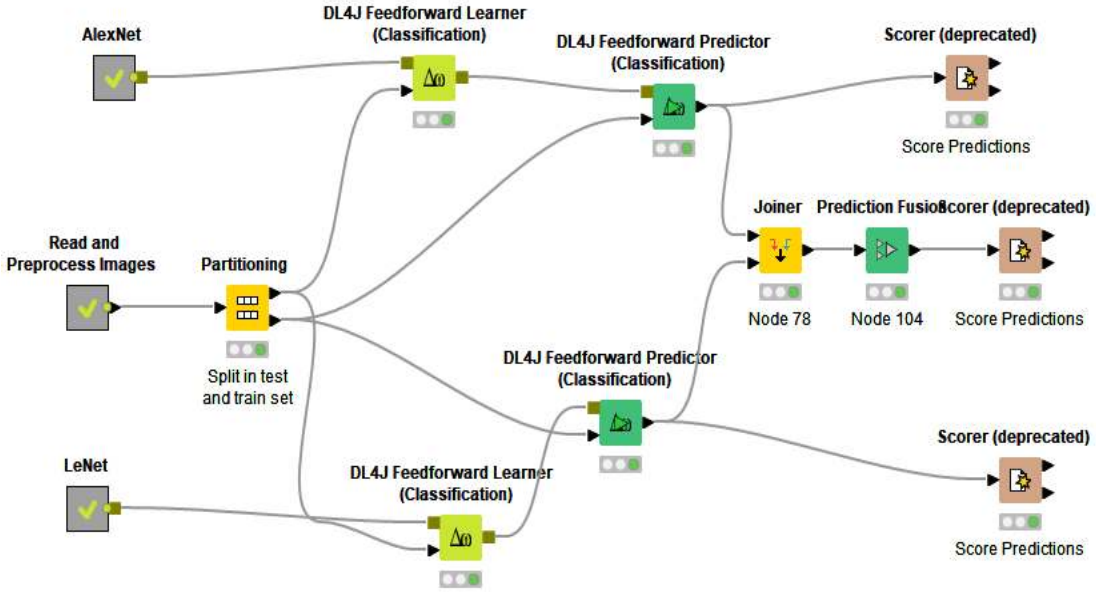
Siteme giren her bir örneği en yüksek oranda doğru sınıflandırıncaya kadar tekrar tekrar çalıştıran bir yöntemdir. Sisteme giren veri setinde yanlış tahmin edilen bir örnek var ise bu örneği başka veya aynı bir veri seti ile aynı veya başka bir algoritmaya yeniden tahmin etmeye çalışır. Şekil 6.4' AdaBoost yöntemi gösterilmiştir.



Şekil 6.4. AdaBoost [47]

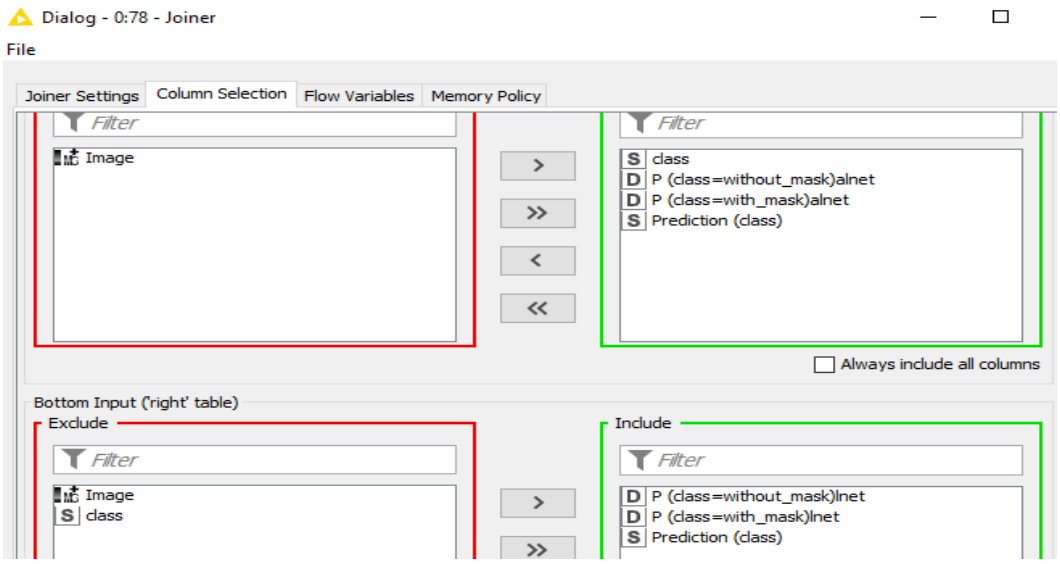
6.3. Knime'da Ensemble Learning

Bu tez çalışmasında üzerinde çalışılan derin öğrenme uygulamasında düğümlerin uçtan uça bağlanarak oluşturduğu iş çalışma akışı şekil 6.5'de gösterilmiştir. Resimleri "Read File" okuyup daha sonra işleyen Knime resimleri "Partitioning" düğümüne ayarlanan oranda ki burada %80 eğitim %20 test için ayrılmıştır. Veri setinin eğitim için ayrılan %80lık kısmı DL4J Feedforawrd Learner(Classification) düğümüne verildi. İki girişli olan bu düğümün birinci girişine derin öğrenme mimarisinin bitişi bağlanmıştır. Eğitilen verimiz sayesinde artık test verilerinin tahmininde bulunmak için DL4J Feedforawrd Predictor(Classification) düğümüne ihtiyaç vardır. Bu düğüm DL4J Feedforawrd Learner(Classification) düğümümden gelen sonuçları birinci girişinden ve test için ayırdığımız veri setinin %20lık kısmını ikinci kısımdan alır. Eğitilen veriden yaralanan DL4J Feedforawrd Predictor(Classification) düğümü bir tahmin sınıfı oluşturur.



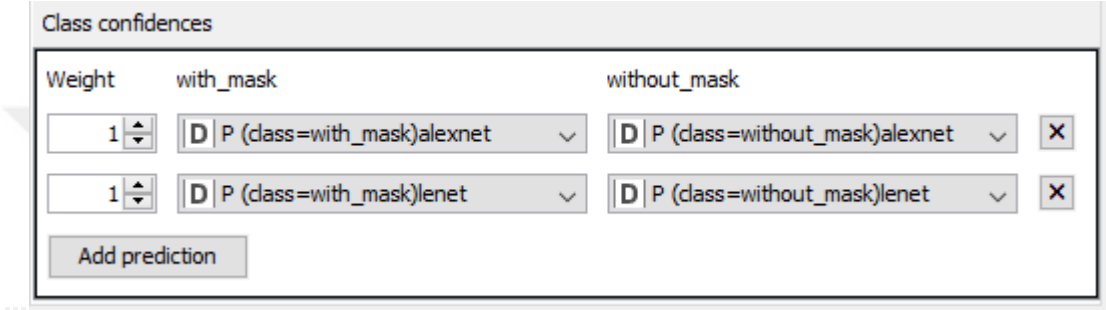
Şekil 6.5. Ensemble Deep Learning Uygulaması

Sıra kullanılan iki farklı mimarinin birleştirilmesine geldi. Bu aşamada "Joiner" düğümünü kullanabiliriz. Tablolar birleştirilirken bir sonraki adımda kullanılacak kolonların seçilmesi işlem kolaylığı sağlar. Şekil 6.6'da tez çalışmamız için gerekli olan kolonlar seçilmesi gösterilmiştir. Kolay kullanım sağlayan Knime'de istenmeyen kolonlar birleştirilmiş toplamaya alınmaz. Bu tez çalışmasında birleştirilmiş kolonda resimlerin bulunması sonucu değerlendirirken zorluk çıkaracağından resim kolonu birleştirilmiş tablodan çıkarıldı. Kullanılan mimarilerin tahmin değerleri ve karşılaştırma işlemi yapılırken kullanılacak olan veri setinin kolonu seçilir.



Şekil 6.6. Knime'de Joiner Düğümü

Knime'de ensemble learning uygulamasında Çoğunluk Oylaması(Majority Voting(MAVL)) yapılabilmesi için "Prediction Fusion" düğümü kullanılmalıdır. Burada kullanılacak tahmin sınıfları ve bu sınıfların hangi mimarilere ait olacağı seçilir. Çoğunluk oylamasında dışarıdan yapılabilecek bir etki olarak doğruluk derecesinin daha çok yüksek olduğu inanılan mimarinin ağırlığı arttırılarak bu mimarinin daha ön planda olması sağlanabilir. Yapılan çalışmada kullanılan mimarilerin ikisi çalışma performansı açısından farklı olmasına rağmen ağırlıklar aynı alınmıştır. Bu da demek oluyor ki her iki algoritmaya aynı şans verildi. Şekil 6.7'de seçilen sınıflar, ağırlıklar ve derin öğrenme mimarileri gösterilmiştir.



Şekil 6.7. Knime'de Prediction Fusion

"Prediction Fusion" düğümünün sonucu şekil 6.8'de gösterilmiştir. Bu şekildeki ilk satır incelendiğinde kullanılan derin öğrenme mimarilerinden LeNet withmask 0 ve AlexNet 0.578 oranında tahminde bulunmuştur. Prediction fusion düğümünde method olarak "mean" yani ortalama değer kullanıldığından bulunan bu iki değer toplanır ve ikiye bölünür. 0 ve 0.578 toplayıp ikiye bölündüğünde sonuç 0.289 çıkar. Aynı şekilde withoutmask olma olasılığı AlexNet 0.422 ve Lnet olasılığı 1 dir. 0.422 ve 1 toplayıp ikiye bölündüğünde sonuç 0.711 çıkar. Sonuçlar prediction fusion olarak kaydedilir ve büyük olan değerin sınıfına dahil edilir. 0.711değeri 0.289 dan büyük olduğu için sonucu withoutmask sınıfına dahil edildi.

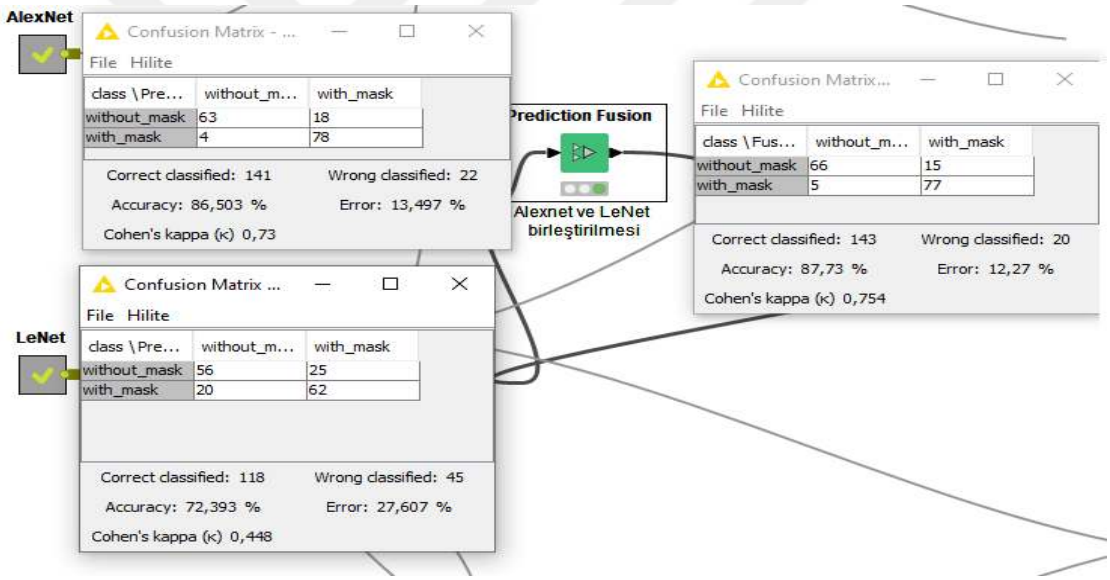
class	P (class=withmask)LeNet	P (class=withmask)AlexNet	P (class=withoutmask)LeNet	P (class=withoutmask)AlexNet	Fused confidence - withmask	Fused confidence - withoutmask	Fused predicti
withoutmask	0	0.578	1	0.422	0.289	0.711	withoutmask
withoutmask	0	0.645	1	0.355	0.323	0.677	withoutmask
withoutmask	0.948	0.01	0.052	0.99	0.479	0.521	withoutmask
withmask	0	0.964	1	0.036	0.482	0.518	withoutmask
withmask	0.001	0.988	0.999	0.012	0.495	0.505	withoutmask
withoutmask	0	0.999	1	0.001	0.5	0.5	withoutmask
withoutmask	0	0.999	1	0.001	0.5	0.5	withoutmask
withmask	0	0.999	1	0.001	0.5	0.5	withoutmask
withmask	0	0.999	1	0.001	0.5	0.5	withoutmask
withoutmask	0	0.999	1	0.001	0.5	0.5	withoutmask
withoutmask	0.001	0.999	0.999	0.001	0.5	0.5	withmask
withoutmask	0.994	0.01	0.006	0.99	0.502	0.498	withmask
withoutmask	0.998	0.01	0.002	0.99	0.504	0.496	withmask
withoutmask	0.998	0.01	0.002	0.99	0.504	0.496	withmask
withoutmask	1	0.01	0	0.99	0.505	0.495	withmask
withoutmask	1	0.01	0	0.99	0.505	0.495	withmask
withmask	1	0.01	0	0.99	0.505	0.495	withmask
withmask	1	0.01	0	0.99	0.505	0.495	withmask
withoutmask	1	0.01	0	0.99	0.505	0.495	withmask
withoutmask	1	0.01	0	0.99	0.505	0.495	withmask
withmask	0.016	0.999	0.984	0.001	0.508	0.492	withmask
withmask	0.017	0.999	0.983	0.001	0.508	0.492	withmask

Şekil 6.8. Fusion Prediction sonucu

7. SONUÇLAR

Veri madenciliği eldeki verilerden yola çıkarak bilgiye ulaşmayı sağlar. Eğitilen eldeki veriler sayesinde tahminler yapılır. Amaç doğruluk değeri olabildiğince yüksek tutmaktır. Bu tez çalışmasında çoğunluğun gücünden yola çıkarak AlexNet ve LeNet derin öğrenme mimarilerinin gücü birleştirilmiştir.

Program sonucunu "Confusion Matrisi" ile kıyaslayalım. Program çalıştırıldığında çoğunluk oylaması kararına göre şekil 7.1'de görüldüğü gibi LeNet %72,393 doğruluk ile oy AlexNet '%86,503 doğruluk payı ile çalışmıştır. Prediction Fusion ise çoğunluk oylaması kararı ile %87,73 doruluk ile iki farklı derin öğrenme mimarisinden çok daha iyi bir sonuç ortaya çıkarmıştır. Bu çalışmada iki farklı derin öğrenme mimarisinin güçlerini birleştirerek kendilerinden çok daha güçlü bir yapı sergiledikleri görüldü.



Şekil 7.1. Karmaşıklık(Confusion) Matrisi sonuçları

KAYNAKLAR

[1] Ş.E. Şeker, Crısp-DM : Endüstriler Arası Standart İş leme – Veri Madenciliği için (Cross Industry Standard Processing – Data Mining), YBS Ansiklopedi. 5 (2018) 10–16.

[2] Z. Bošnjak, O. Grljević, S. Bošnjak, CRISP-DM as a framework for discovering knowledge in small and medium sized enterprises' data, Proc. - 2009 5th Int. Symp. Appl. Comput. Intell. Informatics, SACI 2009. (2009) 509–514. https://doi.org/10.1109/SACI.2009.5136302.

[3] J. Han, Data mining: Data mining concepts and techniques, 2014. https://doi.org/10.1109/ICMIRA.2013.45.

[4] M. Mohammed;, M.B. Khan, E.B. Mohammed, Machine Learning, 2011. https://doi.org/10.1017/cbo9780511974076.010.

[5] Y. Ma, K. Liu, Z. Guan, X. Xu, X. Qian, H. Bao, Background augmentation generative adversarial networks (BAGANs): Effective data generation based on GAN-augmented 3D synthesizing, Symmetry (Basel). 10 (2018). https://doi.org/10.3390/sym10120734.

[6] Ali Ahmet Süzen, K. Kıyas, Derin Öğrenme ve Türkiye'deki Uygulamalar ı, 2018.

[7] A. Habeeb, Introduction to Artificial Intelligence, Res. Gate. 7 (2017). https://doi.org/10.13140/RG.2.2.25350.88645/1.

[8] E. Öztemel, Yapay Sinir Ağlari, 2008. http://papatyabilim.com.tr/PDF/yapay_sinir_aglari.pdf.

[9] Lojistik Resgresyon, (n.d.). https://www.kaggle.com/merveyorulmaz/logistic-regression-diabetes-with-python.

[10] R. Yamashita, M. Nishio, R.K.G. Do, K. Togashi, Convolutional neural networks: an overview and application in radiology, Insights Imaging. 9 (2018) 611–629. https://doi.org/10.1007/s13244-018-0639-9.

[11] A. Zayegh, N. Al Bassam, Neural Network Principles and Applications, Digit. Syst. (2018). https://doi.org/10.5772/intechopen.80416.

[12] P. Kshirsagar, Brain Tumor Classification and Detection using Neural Network, (2020) 83–88. https://doi.org/10.13140/RG.2.2.26169.72805.

[13] M.A. Kızrak, B. Bolat, Derin Öğrenme ile Kalabalık Analizi Üzerine Detaylı Bir Araştırma, Bilişim Teknol. Derg. (2018) 263–286. https://doi.org/10.17671/gazibtd.419205.

[14] F. Kurt, Evrişimli Sinir Ağlarinda HiperParametreleri Etkisini İncelenmesi, Master's Thesis, Eğitim Bilim. Enstitüsü. (2018) 114.

[15] Maliyet Fonksiyonu, (n.d.). https://veribilimcisi.com/2017/07/12/model-olusturma-ve-maliyet-fonksiyonu/.

[16] A. Arı, M.E. Berberler, Yapay Sinir Ağları ile Tahmin ve Sınıflandırma Problemlerinin Çözümü İçin Arayüz Tasarımı, Acta Infologica. 1 (2017) 55–73. dergipark.gov.tr.tr/acin.

[17] S. Shalev-Shwartz, S. Ben-David, Understanding machine learning: From theory to algorithms, 2013. https://doi.org/10.1017/CBO9781107298019.

[18] S. He, S. Ho, S. Ravanbakhsh, Analysis of cosmic microwave background with deep learning, 6th Int. Conf. Learn. Represent. ICLR 2018 - Work. Track Proc. (2018) 1–4.

[19] S.T.D. Merkezi, Derin Farklar Ypaya Zeka Makine Öğrenmesi, (n.d.).

[20] G. Wei, G. Li, J. Zhao, A. He, Development of a LeNet-5 gas identification CNN structure for electronic noses, Sensors (Switzerland). 19 (2019) 1–17. https://doi.org/10.3390/s19010217.

[21] Y. Bengio, I. Goodfellow, A. Courville, Deep learning, Nature. 29 (2019) 1–73.

[22] R. Haldar, R. Chatterjee, D.K. Sanyal, K. Mallick, Deep Learning Based Smart Attendance Monitoring System, (2019).

[23] Ş.E. Şeker, D. Erdoğan, Knime ile Uçtan Uca Veri Bilimi, (2018) 440. https:// sadievrenseker.com /wp-content/uploads/veribilimi_knime.pdf.

[24] https://docs.knime.com/2018-12/deep_learning_installation_guide/index.html#keras-integration

[25] P.T. Singh, The Hotspot Java Virtual Machine : Memory and Architecture, (n.d.) 60–64.

[26] KNIME Deep Learning Integration Installation Guide, (n.d.). https://docs.knime.com/2018-12/deep_learning_installation_guide/index.html.

[27] knimeda feed forward, (n.d.). https://hub.knime.com/knime/extensions/org.knime.features.ext.dl4j/latest/org.knime.ext.dl4j.base.nodes.learn.feedforward.classification.FeedforwardClassificationLearnerNodeFactory.

[28] DL4J Feedforward Predictor (Classification)2, (n.d.). https://hub.knime.com/knime/ extensions /org.knime.features.ext.dl4j/latest/org.knime.ext.dl4j.base.nodes.predict.feedforward.classification.FeedforwardClassificationPredictorNodeFactory.

[29] confusion matrix, (n.d.). https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62.

[30] N. Arizona, Artıfıcıal Neural Networks And Deep Learnıng, Manager. (2001) 43–51.

[31] A. Shrestha, A. Mahmood, Review of deep learning algorithms and architectures, IEEE Access. 7 (2019) 53040–53065. https://doi.org/10.1109/ACCESS.2019.2912200.

[32] A. Sreenivas, M. Maheshwari, S. Jain, S. Choudhary, Indian Sign Language Communicator Using Convolutional Neural Network, 29 (2020) 11015–11031.

[33] J. Mairal, P. Koniusz, Z. Harchaoui, C. Schmid, Convolutional kernel networks, Adv. Neural Inf. Process. Syst. 3 (2014) 2627–2635.

[34] C. Kong, S. Lucey, Take it in your stride: Do we need striding in CNNs?, ArXiv. (2017).

[35] L. Belhaj Salah, F. Fourati, Deep mlp for systems modeling, 2017 18th Int. Conf. Sci. Tech. Autom. Control Comput. Eng. STA 2017 - Proc. 2018-Janua (2018) 49–53. https://doi.org/10.1109/STA.2017.8314891.

[36] E.I. Ünlü, Derin Öğrenme İle Görüntü Bölütleme, 8 (2019) 55.

[37] D. Vikram Mutneja, Methods of Image Edge Detection: A Review, J. Electr. Electron. Syst. 04 (2015). https://doi.org/10.4172/2332-0796.1000150.

[38] T. Tükel, Görüntü İşleme Ve Evrişimsel Sinir Ağları Kullanılarak Diyabetik Retinopati Hastalığının Tespiti, (2019).

[39] Dense Layer, (n.d.). https://hub.knime.com/knime/extensions/org.knime.features.ext.dl4j/latest /org.knime.ext.dl4j.base.nodes.layer.mlp.dense.DenseLayerNodeFactory.

[40] R. Salakhutdinov, G. Hinton, Deep Boltzmann machines, J. Mach. Learn. Res. 5 (2009) 448–455.

[41] F. Doğan, İ. Türkoğlu, Derin Öğrenme Modelleri ve Uygulama Alanlarına İlişkin Bir Derleme, DÜMF Mühendislik Derg. 10 (2019) 409–445. https://doi.org/10.24012/dumf.411130.

[42] Y. Li, J. Gao, Q. Li, W. Fan, Ensemble learning, Data Classif. Algorithms Appl. (2014) 483–509. https://doi.org/10.1201/b17320.

[43] Y. Chen, Y. Wang, Y. Gu, X. He, P. Ghamisi, X. Jia, Deep Learning Ensemble for Hyperspectral Image Classification, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 12 (2019) 1882–1897. https://doi.org/10.1109/JSTARS.2019.2915259.

[44] M. Koptur, Makine Öğrenimi, (n.d.). https://makineogrenimi.wordpress.com/2017/06/14/ modellerin-birlestirilmesi-ensemble-learning/.

[45] M. Koptur, Makine Öğrenimi4, (n.d.). https://makineogrenimi.wordpress.com/2017/06/19/ modellerin-birlestirilmesi-ensemble-learning-4/.

[46] R. Madan, Bagging machine Learning, (n.d.). https://medium.com/analytics-vidhya/how-to-use-bagging-technique-for-ensemble-algorithms-a-code-exercise-on-decision-trees-6f944cacf085.

[47] M. Koptur, Makine Öğrenimi Adaboost, (n.d.). https://makineogrenimi.wordpress.com/2017/06/19/modellerin-birlestirilmesi-ensemble-learning-4/.

ÖZGEÇMİŞ

Hilal ÇELİK

[redacted]

ARAŞTIRMA DENEYİMİ

- ✓ KNİME, PYHTON, MİCROSOFT OFFİCE
- ✓ Programlama-Yazılım Geliştirm: Java, JSP, Python
- ✓ Web Programlama : Html,CSS,Javascript
- ✓ Veri Tabanı : MySQL,MsSQL