YAPAY ZEKA VE YÖNTEMLERİ

2021 Bahar Dönemi
PROJE 2

05170000021 - Enes Alper BALTA
05170000057 - Ramazan EYMİR

İçindekiler

1.Problemin Tanımı

Yapay Zeka Yöntemleri Proje 2 için yapmayı seçtiğimiz konu proje açıklamasında verilen 6 numaralı konu olan 'Doğal Dil İşleme' alanındadır. Konuyla alakalı *Sentiment Analysis* yapan bir *Deep Learning* modeli kurmaya karar verdik.
Bu konu altında yaptığımız araştırmalarda bir çok datasete gözatmış olduk. Bu datasetlerin genellikle büyük boyutlara sahip olduğunu gözlemledik. Örnek olarak vermek gerekirse 9 GB'lık Yelp Review ve yaklaşık 230 milyon inceleme içeren Amazon yorumları gibi datasetleri inceledik. Bu durum lokalde gerçekleştireceğimiz çalışmalarda hem eğitim süresini hem test süresini çok fazla uzatacağı için daha az sayıda veri içeren bir dataset kullanmaya karar verdik. Bu dataset, kullanıcılar tarafından Twitter üzerinde atılmış tweet'leri içermekte. Bu tweetler sexist/racist yorum içermesine göre işaretlenmiş durumda.
Datasetimizden bağımsız olarak, sosyal medya sitelerinde kullanıcılar tarafından günde milyonlar içerek paylaşılmakta. Bu site herhangi bir fotoğraf paylaşım sitesi olsa bile yazılı bir metin içermekte. Bu siteler/sistemler kullanıcıların paylaşabileceği veya yorum yapabileceği alanlarda hakaret, aşağılama gibi yazıları engelleyen bir filtrelemeye sahip olmalı. Bizim bu dataset ile yapmak istediğimiz, dataseti kullanarak böyle bir uygulamanın kullanıcılarının yazmış olduğu metinleri sınıflandırmaktır. Bu sınıflandırma sonucunda bu kullanıcılara karşı bir aksiyon alarak uygulamanın güvenliği sağlanabilir.

2.Araştırma (Ön Çalışma)

Gerçekleştireceğimiz projemizin konusunu kesinleştirmeden önce derin öğrenme alanında bir proje gerçekleştireceğimize karar vermiştik. Bu nedenle hem ders notları içerisinde bulunan hem de internet üzerinden rahatlıkla erişebileceğimiz MIT Deep Learning 6.S191 derslerinden büyük şekilde yararlandık.

Derin öğrenme konusunda başlangıç için temel bilgilerini edindikten sonra Natural Language Processing alanında projemizi gerçekleştirmeye karar verdik. Bu bağlamda TensorFlow ve Keras üzerindeki kod örneklerine göz gezdirdik ve birçok konuda bu örneklerden yararlandık.

Tekrar incelediğimiz ders notlarınız, internet üzerinden dinlediğimiz MIT Deep Learning dersi ve incelemiş olduğumuz kod örnekleri, bizi Deep Learning alanındaki bilgi dağarcığımızı daha fazla arttışmış olup, gerçekleştirdiğimiz NLP projesi ile de bu bilginin yoğurulmasında çok büyük bir katkı sağlamıştır.

3.Kullanılan Ortam, Yöntem ve Kütüphaneler

Projemizi ortak bir şekilde çalışmanın verimli bir şekilde gerçekleştirilebildiği Google Colaboratory ortamı üzerinde birlikte çalışarak tamamladık. Projemizi Deep Learning alanında gerçekleştirmeye karar verdikten sonra beraberinde getirdiği gerekli tool'lar nedeniyle TensorFlow kütüphanesi ile çalışmaya karar verdik. TensorFlow içerisinde bulunan Keras kütüphanesi altındaki Derin Öğrenme modellerden projemize uygun olan ve sıralı katmanlardan oluşması sebebiyle kavraması daha basit olan 'Sequential' modelini kullandık.

Bu model ile birlikte kullandığımız framework ve sürümleri ise;

Numpy	-	1.19.5
Pandas	-	1.1.5
Matplotlib	-	3.2.2
TensorFlow	-	2.5.0
PIL	-	7.1.2
NLTK	-	3.2.5
WordCloud	-	1.5.0
TextAugment	-	1.3.4

4.Önerilen (Geliştirilen/Kullanılan) Yöntem

Bir sonraki başlıkta değinilecek olan veriseti gözlemlerinin dengeli dağılmaması ve yetersiz gözlemin bulunması problemini çözdükten sonra WordCloud kütüphanesini kullanarak bir görselleştirme işlemi gerçekleştirdik. Bu işlemi gerçekleştirmekteki amacımız yeni oluşturduğumuz veriseti içerisindeki frekansı yüksek kelimeleri gözlemlemek ve verisetimizin genel durumuyla alakalı kolay bir çıkarım gerçekleştirebilmektir.

Görselleştirme sonrasında model oluşturma yolundaki ilk adımımız olan ve TensorFlow kütüphanesi altında bulunan TextVectorization işlemini, tüm veri setimiz üzerinde gerçekleştiriyoruz. Bu işlem verdiğimiz parametrelere göre belli bir veri tipi türünde her bir kelimenin eşsiz temsilini gerçekleştirmektedir. Bu temsil işlemi de belirli bir liste içerisinde biriktirilmektedir. Böylece veri setimiz içerisinde bulunan kelimelerin temsili gerçekleştirilmiş olur. Son oluşan yapıyı gerçek hayattaki sözlük gibi düşünebiliriz.

Vektörize edilmiş olan veri setimizi Train, Validation ve Test veri setlerine belirli yüzdeliklerle ayırma işlemini gerçekleştiriyoruz. Verktörize edilmiş olan veri setimiz, TensorFlow'un bir veri tipi olan 'MapDataset' tipinde bulunduğu için 'take' ve 'skip'

metotlarıyla önceden belirlemiş olduğumuz yüzdelerle veri setimizi kolaylıkla ayrıştırabiliyoruz.
Eğitim süresini ve önişleme süresini optimize etmek amacıyla TensorFlow dökümanlarında belirtildiği gibi train, validation ve test veri setlerimize 'Prefetching' işlemini gerçekleştiriyoruz.
Yukarıda gerçekleştirilen işlemleri tamamladıktan sonra oluşturduğumuz veri seti ile oluşturacağımız iki farklı modeli test ediyoruz.
İlk modelimiz;

- Daha önce de bahsedildiği gibi modelimiz TensorFlow içerisinde bulunan Keras kütüphanesi altındaki Deep Learning modeli olan Sequantial model ile oluşturulmuştur.
- İçerisinde sıralanmış olan katmanlar ise şöyledir:
- **Embedding**
    - Veri setimiz yazılardan oluştuğu için ilk katman olarak Embedding katmanını belirledik.
    - Bu katman her biri eşsiz Integer değerleriyle temsil edilen yazıları kullanmaktadır. Bu işlemi de veri setimizi vektörize ederek gerçekleştirmiştik.
    - input_dim, metin verilerindeki kelime dağarcığının boyutudur. Vektörize işlemi gerçekleştirirken bu boyutun 0 ile 10000 arasında olacağını belirttiğimiz için 10001 olarak parametremizi veriyoruz.
    - output_dim, kelimelerin gömüleceği vektör uzayının boyutudur. 16 olarak belirlenmiştir.
- **Dropout**
    - Overfitting durumunu önlemek adına tüm ağ içerisinden rastgele seçilecek %20 nöron setinin eğitim aşamasında göz ardı edilmesini sağlamaktadır.
- **GlobalAveragePooling1D**
    - Global Average Pooling işlemini gerçekleştirmektedir. Bu işlem 3D tensor olarak gelen verileri 2D tensor olarak döndürmektedir.
- **Dense**
    - Densely-connected olan sinir ağı katmanıdır.
    - Bu katman sonrasında verilerimizin çıktı değerlerini alırız.
    - Çıktı değerleri, bir sonraki başlıkta belirteceğimiz loss fonksiyonu olan BinaryCrossentropy nedeniyle sadece bir değer olarak oluşacaktır. Bu sebeple de Dense katmanını tek boyutlu oluşturuyoruz.

- Modelimizi oluşturduktan sonra compile işlemini gerçekleştirirken verdiğimiz parametreler ise;
- **loss**
    - BinaryCrossentopy, gerçek değerler ve tahmin edilmiş değerler arasındaki çapraz entropi kaybını hesaplar.
    - Bu yöntemi genelde ikili sınıflandırma uygulamalarında fazlasıyla kullanıldığı için tercih etmiş bulunmaktayız.
- **optimizer**
    - Sıkça kullanılan ve gradient descent yöntemini kullanan 'adam' optimizerını kullandık.
- **metrics**

    - Projemizde ikili etiketlerin tahminlenme durumu bulunduğu için metrik yöntemi olarak BinaryAccuracy kullandık.

Oluşturulan ilk modelin train veri seti üzerinde 5 epoch çalışması sonucundaki değerlerimiz:

loss : 0.1041 **binary_accuracy** : 0.9600
val_loss : 0.1359 **val_binary_accuracy:** 0.9492

Oluşturulan ilk modelin test veri seti üzerinde çalışması sonucundaki değerlerimiz:

loss : 0.1384 **binary_accuracy** : 0.9430

İkinci model üzerindeki gerçekleştirdiğimiz değişiklikler ise;

- **Dense**
    - Son katman olan Dense katmanına aktivasyon fonksiyonu olarak **'relu'** fonksiyonunu belirledik.
- **metrics**
    - BinaryAccuracy değerinin opsiyonel olan **threshold** parametresine 0.25 değerini atadık.
    - Threshold, tahmin değerlerinin 1 mi yoksa 0 mı olduğuna karar vermek için kullanılan float tipinde bir eşik değeridir.

Değişiklikler sonucunda oluşturulan ikinci modelin train veri seti üzerinde 5 epoch çalışması sonucundaki değerlerimiz:

loss : 0.1592 **binary_accuracy** : 0.9682
val_loss : 0.2086 **val_binary_accuracy:** 0.9502

Değişiklikler sonucunda oluşturulan ikinci modelin test veri seti üzerinde çalışması sonucundaki değerlerimiz:

loss : 0.2286 **binary_accuracy** : 0.9519

Görüldüğü üzere loss değerlerinde bir artış görülmesine rağmen, accuracy değerlerinde de bir miktar artış izlenmiştir.

5.Deneysel Çalışmalar

En son karar kıldığımız Twitter Sentiment Analysis datasetinden önce çok fazla dataseti araştırdık. Bu yüzden bunları paylaşmak istiyoruz.

1. Cornell Üniversitesi farklı tiplerde birçok kaynak paylaşmakta. Bu datasetlere bakıldığında datasetlerin boyutlarının çok üst seviyelerde olduğunu görüyoruz, 13, 24GB gibi. Cornell Üniversitesinin datasetlerine buradan ulaşabilirsiniz. Bu site üzerinde sunulan duygu analizi dataseti olan Movie-Review-Data datasetine baktığımızda ise, bu dataset tek bir dosya yerine +1000 txt dosyasından oluşmakta. Çok karışık geldiği için bu dataseti kullanmadık.
2. 3 tane daha dataseti detaylı bir şekilde inceledik. Bunlar Yelp Dataset, herkes tarafından bilinen Sentiment140, ve Amazon Review datasetleri. Bu datasetler, ilk karar verdiğimiz andan itibaren, proje konusu olarak belirlediğimiz NLP ve

duygu analizi alanında sektörde çok fazla bilinen ve üzerine birçok çalışma yapılmış datasetler. Bu datasetleri seçtiğimizde çıkıcak sıkıntı ise, en küçük dataset olan Sentiment140 dataseti bile +200MB olduğu için basit bir Deep Learning modelinin bile bir sonuç vermesi çok zaman alıcaktı. Bu yüzden bu datasetleri seçmemeye karar verdik.

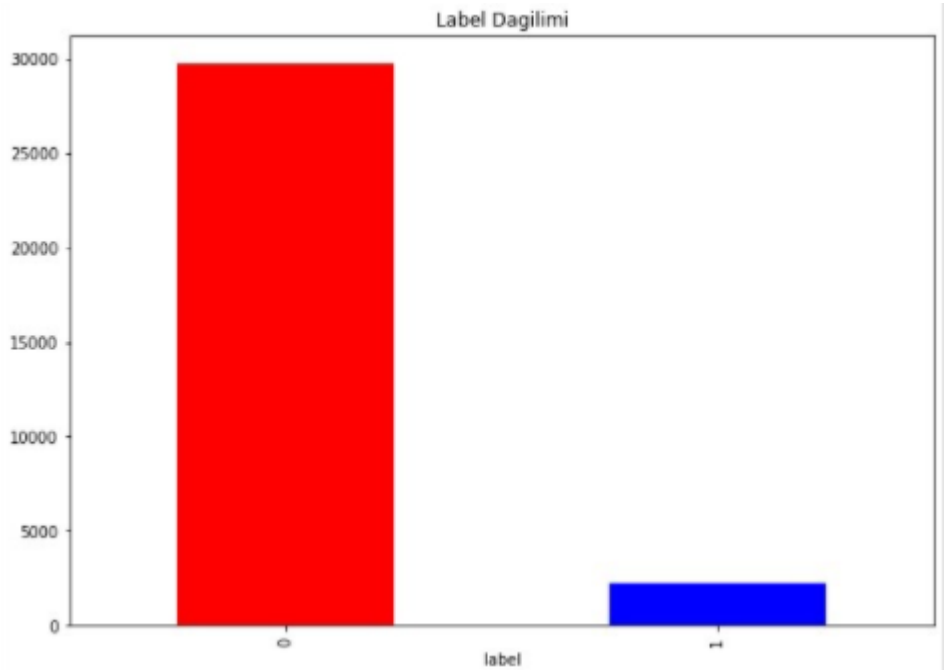
Seçilen Dataset
Kullandığımız dataset, Medium.com gibi sitelerde yazılmış olan 'Datasets for Sentiment Analysis' gibi yazıları okurken karşımıza çıktı. Daha sonra Kaggle'a girerek bu dataset hakkında detaylı bilgi edindik. Dataset aslında bir kullanıcı tarafından, test dataseti üzerinde sınıflandırma yaparak, kullanıcılar arasında bir yarışma olması açısından paylaşılmış. Biz bu test dataseti üzerinde, sınıflandırmalar olmadığı için, programımız içerisinde sadece train dataseti üzerinde işlem yaptık. Datasetin linkine buradan ulaşabilirsiniz.

Veri Seti Özellikleri
Daha önce belirttiğimiz üzere dataset olarak sadece train.csv dosyasını kullandık. Bu dosya içerisinde 31962 satır veri içermekte. Bu veriler id, label ve tweet niteliklerini içermekte. Id, sıralamaya göre index olarak kullanılmakta ve programımız içerisinde bir işlevi olmadığı için, veri okunduktan sonra droplanmaktadır. Tweet'ler sexist/racist olmasına göre sexist/racist ise 1, değilse 0 olarak belirlenmiş durumda.
Bu datasetimizde 2 tane sınıf olduğunu göstermekte ve bu proje içerisinde binary sınıflandırma yapmamızı gerektirmektedir.

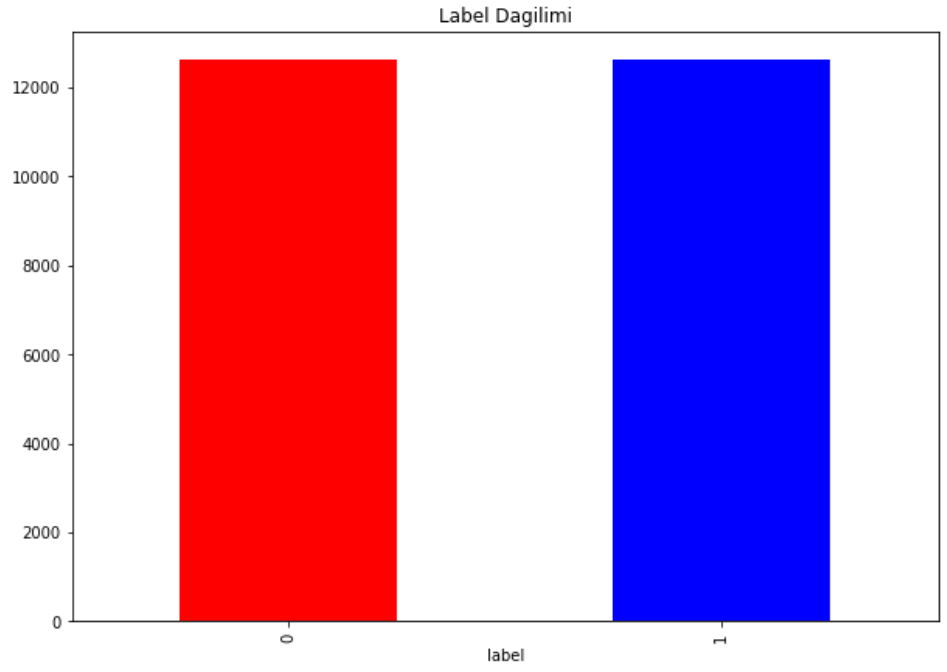
Dataset Üzerinde Yapılan Çalışma
Dataset üzerinde karşılaştığımız en büyük sorun 0 ile işaretlenmiş tweet'lerin 1 ile işaretlenmiş tweet'lere göre yüzdesel olarak çok fazla bulunması. 0 ile işaretlenmiş yaklaşık 30 bin veri bulunmasına karşın, 1 ile işaretlenen veri sayısı 2500 civarlarında. Ara raporda da belirttiğimiz üzere bu dataset üzerinde resampling yapılması gerekiyor. Yaptığımız araştırmalar sonucunda dataset üzerinde 'Undersampling' ve 'Oversampling' işlemlerinin yapılabildiğini gördük. Undersampling işlemine göre, 0 ile işaretlenen 30 bin civari data içerisinden 1 ile işaretlenen data sayısı kadar rastgele alarak bunları eşitliyoruz. Oversampling işleminde ise, 1 ile işaretlenen 2500 datayı kopyalayarak 30 bine getirerek bu sefer, dataseti bu şekilde eşitliyoruz. Bunların ikisini de kullanabilmek için 0 ve 1 olarak işaretlenmiş data sayısını yaklaşık 13 bin civarında eşitledik.
Bu 13 bin sayısını rastgele belirleyerek data sayılarını ayarlamadık. Oversampling işlemi için internette yaptığımız araştırmalar sonucunda 'Data Augmentation' adlı işlemi yapmamız gerektiğini gördük. Daha sonra bu işlemi yapan kütüphaneleri araştırdık ve şu kütüphaneleri bulduk: Facebook'un çok yakın zamanda çıkardığı Augly, NlpAug ve TextAugment kütüphanesi. Augly kütüphanesi, text augmentation işlemlerini daha çok karakter bazlı yaptığı için onu kullanmak istemedik. NlpAug ve

TextAugment kütüphanelerini kelime bazlı değişiklik yapan fonksiyonlarına göre incelediğimizde ise, TextAugment kütüphanesinin çok daha iyi sonuçlar verdiğini gördük. Bu yüzden projemizde TextAugment kütüphanesini kullandık. Bu kütüphane içerisindeki 6 tane fonksiyonu, train datasetimiz içerisindeki 1 label'ina sahip tweet'lere uyguladık. Bunun sonucunda yaklaşık 13 bin tane verimiz oluşmuş oldu. 0 ile işaretlenmiş 13 bin tane veriyi de almış olduğumuzda 26 bine çok yakın sayıda satır içeren bir dataset elde etmiş olduk.

Dataset'in Önceki Hali



Dataset'in Güncel Hali



Data sonra bu dataseti %64 eğitim, %16 validation ve %20 test verisi olmak üzere parçaladık.

6. Sonuç

Proje ara raporda yapılan haliyle, datasetin dengesiz olmasından dolayı, her ne kadar accuracy değerleri %90'ların üzerinde olsa da, %90'ın üzerinde 0 label'ina sahip data ile bu işlemi yaptığımız için bir nevi overfitting durumu yaşamış olduk. Bu durumu düzeltmek için yukarıda anlatılan işlemleri yaptığımızda, kurduğumuz modeller ile test verisinin tahminlenmesi sonucu %94.30 ve %95.19 gibi sonuçlar aldık ve herhangi bir overfitting durumuyla karşılaşmadık. Raporun giriş kısmında da belirtildiği gibi, Twitter üzerinde atılan bir tweetin sexist/racist bir yorum içermesini %95 gibi bir oranla doğru tahmin ederek, eğer içeriyorsa bu tweet'in paylaşılmaması, kaldırılması, kullanıcının hesabının kapatılması gibi işlemler yapılarak yüksek bir başarı elde edilmiş olur.
Bu proje Twitter üzerinde bir üstte bahsedilen konularda, gerçek hayatta Twitter yönetiminin işine yarayabilir. Kullanıcılara böyle tweetleri daha az göstererek, daha yüksek kalitede bir kullanıcı deneyimi sağlayabilir. Aynı şekilde kullanıcılar da, kendileriyle alakalı olsun olmasın böyle bir yorumu görmeyerek daha sağlıklı bir psikolojiye sahip olabilir. Her ne kadar bu projeyi Twitter'i örnek göstererek açıklasakta, böyle bir fıltreleme yöntemiyle herhangi bir sosyal platform bu tarz bir proje ile sistemlerini geliştirebilir. İnsana yararı, böyle kötü yorumları görmeyerek, internet üzerinde daha kaliteli zaman geçirebilir ve herhangi bir mental/psikolojik sorunla karşılaşma riski azalır.
Bu projeyi geliştirirken Deep Learning aşamasına gelmeden önce, kelimeleri buna hazır hale getirebilmek için kullanılan NLTK hakkında bilgi sahibi olduk. Özellikle NLP aşamasında hangi işlemlerin yapıldığını, word tokenize, text vectorization gibi birçok yeni şey öğrendik. Tüm bunların dışında dataset üzerinde, raporun 5.bölümünde yapılan işlemler ile, daha önce hiç kullanmadığımız ve bilgi sahibi olmadığımız bir alanda yani data augmentation alanı hakkında bilgi edindik ve kullandık. TensorFlow ile kurduğumuz modelde karşılaştığımız, modelin oluşturulması, inputların ayarlanması, shape hataları gibi, sorunları çözerek bir çok farklı bilgi edindik.
Genel olarak bu proje bize NLP, Deep Learning ve TensorFlow hakkında bir çok yeni şey öğretti.9430 9519

Ek 1

Başarıyı arttırmak için model üzerinde activation fonksiyonlarda ve threshold değerlerinde değişiklikler yaptık. Test verisi üzerinde %94.30 olan başarıyı %95.19'a yükselttik. Bu model işlemlerini yaparken bir çok farklı yöntem kullanarak model

eğitimi işlemini gerçekleştirdik. Fakat en yüksek doğruluk değerini bu işlemleri yaparak elde ettik.

Ek 2

Projemiz, konumuz olan NLP, deep learning ve bunlardan oluşan duygu analizi anlamında yapılan çalışmalara oldukça benzemektedir.

Ek 3

NLP için Şadi Evren Şeker'in verdiği makine öğrenmesi kursunun içerisindeki NLP bölümünden, kelimelerin nasıl işleneceği ile alakalı bölümden faydalandık. Burada olmayan farklı yöntemler ekleyerek dil işleme bölümünü daha etkin bir şekilde gerçekleştirdik. Deep learning alanında ise kaggle/keras/tensorflow gibi sitelerin öğreticilerinden faydalanıp, diğer kişilerin yazdığı kodlara göz atıp anlamaya çalıştık. Bunların sonucunda da, doğal dil işlemeyle alakalı işlemleri yapıp kendi modelimizi oluşturduk. Bu kullandığımız kaynaklarda tam olarak bizim yaptığımız işlem olmadığı için, genel olarak baktığımızda kodumuz bu sitelerdekilere göre çok farklı işlemler içermektedir.

Kaynakça

1. http://introtodeeplearning.com/
2. https://www.tensorflow.org/
3. https://github.com/facebookresearch/AugLy
4. https://www.google.com/search?q=nlpaug&oq=nlpaug&aqs=chrome.0.69i59j46i175i199j0i30l4j69i60j69i61.712j0j7&sourceid=chrome&ie=UTF-8
5. https://github.com/dsfsi/textaugment
6. https://towardsdatascience.com/
7. https://www.kaggle.com/
8. https://bilgisayarkavramlari.com/
9. https://bilkav.com/makine-ogrenmesi-egitimi/
10. https://learn.datacamp.com/courses/deep-learning-in-python

Özdeğerlendirme Tablosu

	İstenen Madde	Var	Açıklama	Tahmini Not
1	Kapak Sayfası, Problemin Tanımı, Kullanılan Ortam, Yöntem ve Kütüphaneler(10)	Yapıldı	Tüm istenilen maddeler anlaşılır bir şekilde açıklandı	10
2	Araştırma(10)	Yapıldı	Çok kapsamlı ve detaylı bir araştırma yapılarak konular hakkında detaylı bilgi edinildi.	10
3	Önerilen Yöntem(10)	Yapıldı	Yöntem, raporda detaylı bir şekilde açıklandı, yüksek bir doğruluk değeri yakalandı.	10
4	Deneysel Çalışmalar(10)	Yapıldı	Veri seti bulmak için detaylı araştırma yapıldı, mevcut dataset üzerinde kapsamlı araştırma ve işlemler yapıldı.	10
5	Proje Rapor Biçimi, Organizasyonu, Boyutu, Kalitesi, Kaynakça ve atıflar(10)	Yapıldı	Rapor özenle istenilen şekilde hazırlandı. Kullanılan kaynaklar listelendi.	10
6	Sonuç(10)	Yapıldı	Sonuçlar hakkında detaylı açıklama yapıldı, gerçek hayatta kullanılabilirliği ve insalara yararı açıklandı.	10
7	Ek 1: Başarım İyileştirme(10)	Yapıldı	İlk modelimizde başarımız oldukça yüksek olduğu için çok daha az bir başarı iyileşmesi gözlemlendi fakat bu modeli oluşturmak için bile bir çok deneme yapıldı.	8
8	Ek 2 (10)	Yapıldı	Literatür tarandı fakat çok büyük bir katkı sağlanamadı.	8
9	Ek 3 (10)	Yapıldı	Detaylı açıklama sağlandı.	10
10	Özdeğerlendirme Tablosu(10)	Yapıldı	Yapılanlar düzgün ve doğru bir şekilde açıklandı	
			100 Üzerinden Toplam Not	96