

CNN Derin Öğrenme Modeli ile Beden ve Ürün Tavsiye Sistemi Oluşturma

Gizem KARATAŞ



Proje Konusu



Kullanıcı tarafından alışveriş sitesine yüklenen fotoğrafın uygun olup olmadığını tespitinden sonra görüntü işleme ile beden ölçülerini tahminleyerek kullanıcının incelediği ürünler için beden tavsiyesinde bulunmak, yine görüntü işleme ile vücut tipini tahminleyerek kullanıcının vücut tipine uygun kıyafet önerilerinde bulunmak ve son aşama olarak beğendiği ya da tavsiye edilen kıyafetleri yüklediği fotoğrafa işleyerek kıyafeti kendi üstünde görme fırsatı sağlamaktır.

Proje Hedefi

Proje hedefi, müşteri memnuniyetini artırmak, iade oranları düşürmek, müşterinin karar verme ve satın alma hızı artırmak, yenilik içeriği için dikkat çekmek ve dolayısıyla bu da site ziyaretlerini ve satışı artırmak hedeflenmektedir.



Proje Adımları



1. Adım

Yüklenen fotoğrafın insana ait olup olmadığıının tespiti

2. Adım

Yüklenen fotoğrafta işaretleme yöntemi ile beden ölçülerinin tahmin edilmesi

3. Adım

Yüklenen fotoğraf analiz edilerek vücut tipinin tahmin edilmesi

4. Adım

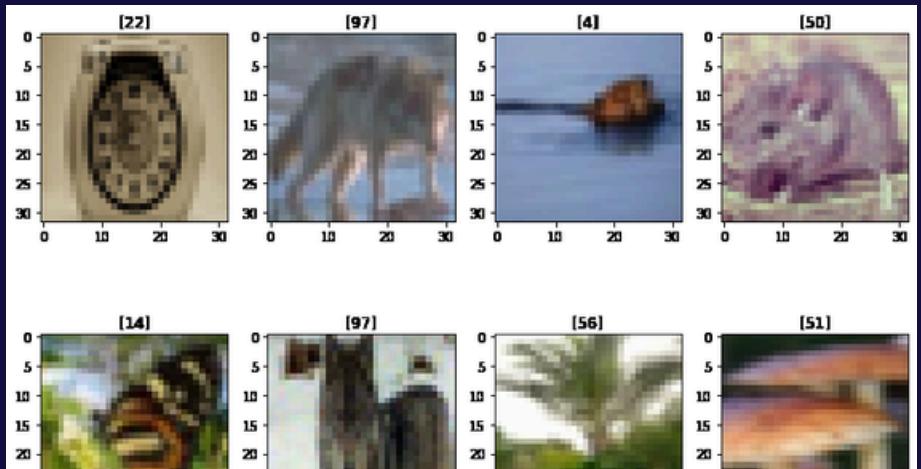
Tavsiye edilen ürulen yüklenen fotoğrafın birleştirilerek müşterinin ürünü direkt kendi üstünde görmesinin sağlanması.

1. Adım

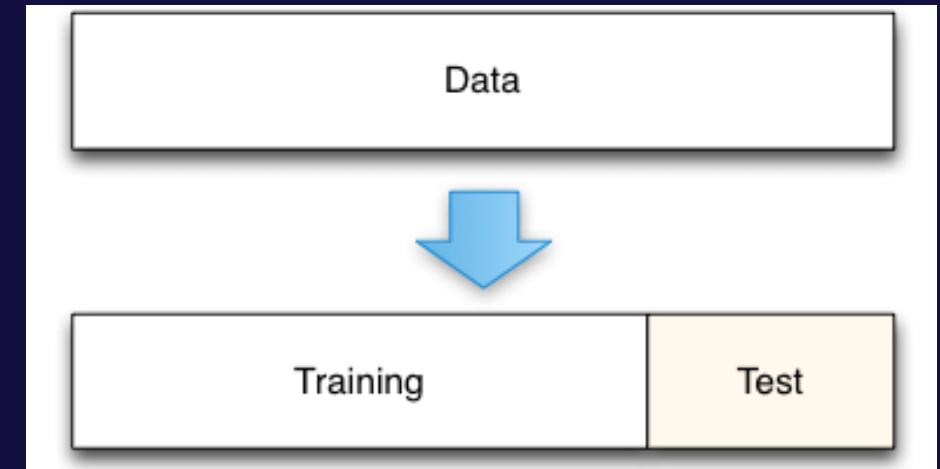
Bu projede Ana Projenin İlk Adımı olan CNN Derin Öğrenme Modeli ile Kullanıcının Yüklediği Fotoğrafın Uygunluğu Değerlendirilmesini gerçekleştirildi.



Cifar100 Veri Seti



[apple]	→ apple,
[beetle]	→ beetle,
[bicycle]	→ bicycle,
[bottle]	→ bottle,
[bowl]	→ bowl,
[boy]	→ boy,
[bridge]	→ bridge,
[bus]	→ bus,
[butterfly]	→ butterfly,
[camel]	→ camel,
[can]	→ can,
[castle]	→ castle,
[caterpillar]	→ caterpillar,
[cattle]	→ cattle,
[chair]	→ chair,
[chimpanzee]	→ chimpanzee,
[clock]	→ clock,
[cloud]	→ cloud,
[cockroach]	→ cockroach,
[couch]	→ couch,
[crab]	→ crab,
[crocodile]	→ crocodile,
[cup]	→ cup,
[dinosaur]	→ dinosaur,
[dolphin]	→ dolphin,
[elephant]	→ elephant,
[flatfish]	→ flatfish,
[forest]	→ forest,
[fox]	→ fox,
[girl]	→ girl,
[hamster]	→ hamster,
[house]	→ house,
[kangaroo]	→ kangaroo,
[computer keyboard]	→ computer keyboard,
[lamp]	→ lamp,
[lawn-mower]	→ lawn-mower,
[leopard]	→ leopard,
[lion]	→ lion,
[lizard]	→ lizard,
[lobster]	→ lobster,
[maple tree]	→ maple tree,



Boyutu

Renkli 32x32 piksel
görüntüden oluşur

Özellikleri

İçinde 100 farklı sınıf'a ait
görsel bulunur. Genellikle
hayvanlar, bitkiler, insanlar,
araçlar, mobilyalar gibi
çeşitli kategorileri temsil
eder.

Spliti

Eğitim seti, 50.000
görüntüden oluşur, her
sınıf için 500'er görüntü
icerir. Test veri seti,
10.000 görüntüden oluşur,
her sınıf için 100'er
görüntü içerir.

Metod:

Cifar100 veri seti çok fazla sınıf barındırdığı için bu kadar sınıf ile eğitilen modelin spesifik tek bir alt ya da üst sınıfı tahminleme doğruluk oranının çok düşük olmasını bekleriz, hem eğitilmesi uzun zaman almaktadır hem de sonuçları güven vermeyecek kadar düşük olacaktır. Bu sebeple veri seti projenin amacı doğrultusunda yeniden sınıflandırıldı. Projenin ilk adımındaki amaç kullanıcı tarafından yüklenen görselin insan olup olmadığına tahminlemesini yapmaktadır. Bu sebeple veri seti aşağıdaki şekilde bölünmüştür:

1 : İnsan Olanlar ==> [2, 11, 35, 41, 46, 98] kız bebek, erkek bebek, kız çocuk, erkek çocuk, kadın, erkek

0 : İnsan Olmayanlar ==> insan labelları haricindeki tüm label etiketleri

bu şekilde ikiye ayrılan veri setinin insan olan ve olmayan olarak sınıflandırarak doğruluk oranının çok daha yüksek çıkışını sağlamıştır. İnsan verileri ve diğer veriler arasındaki dengesizlik giderilerek yüksek tahmin kabiliyeti elde edilmiştir.

CNN Derin Öğrenme Modeli

Özellikle görsel verilerle çalışmak için tasarlandığı için bu projede CNN derin öğrenme modeli kullanılmıştır.

CNN, görsel veri analizi için tasarlanmış bir yapay sinir ağları türüdür. Görüntü işleme problemlerinde başarılı olan bu model, özellikle görüntü sınıflandırma, nesne tanıma, yüz tanıma, çizgi çekme gibi görevlerde kullanılır. CNN'ler, verilerdeki hiyerarşik özelliklerini öğrenme yeteneğine sahiptir.

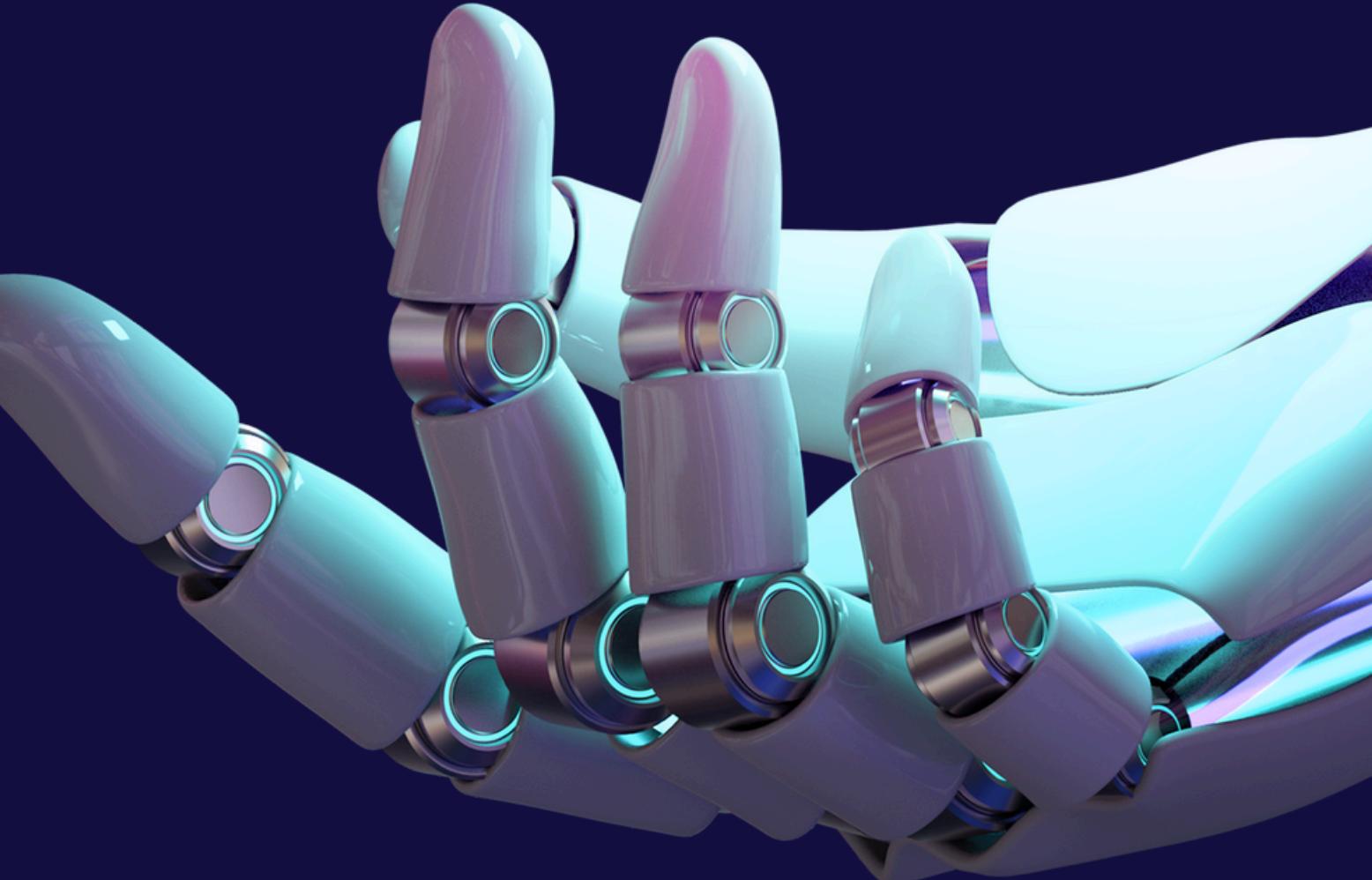
Bölgelerin Özelliklerinin İncelenmesi: CNN'ler, görüntülerin farklı bölgelerindeki özellikleri öğrenebilir. Bu, nesne tespiti gibi görevler için gereklidir. Karmaşık Modelleme: Görüntüler genellikle karmaşık yapılar içerir. CNN'ler, bu karmaşıklığı öğrenebilir ve yüksek doğrulukla tahminlerde bulunabilir.

Yapısal Özelliklerin İncelenmesi: Görüntülerdeki yapısal özellikler (kenarlar, köşeler, desenler vb.) CNN'ler tarafından öğrenilebilir. Bu özellikler, nesne tanıma ve sınıflandırmada önemlidir. Ölçek Uyumlu Modelleme: CNN'ler, görüntülerdeki özelliklerin ölçekte değişikliklerine karşı dayanıklıdır. Bu, farklı boyutlardaki nesneleri tanıtmak için idealdir.

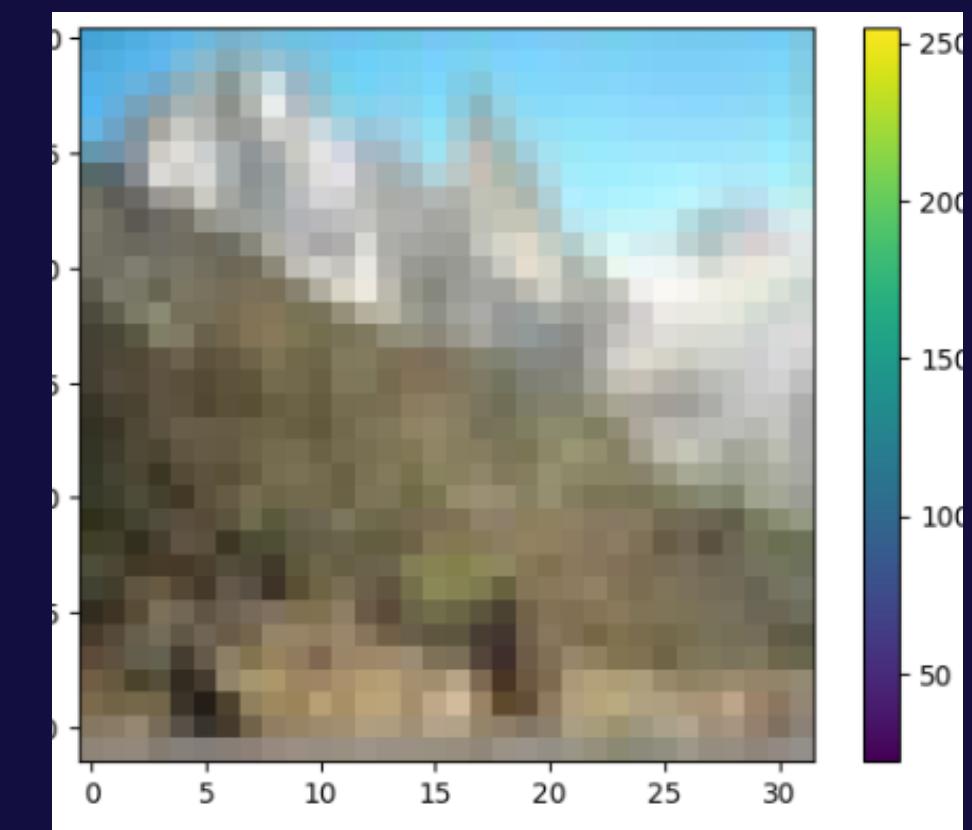
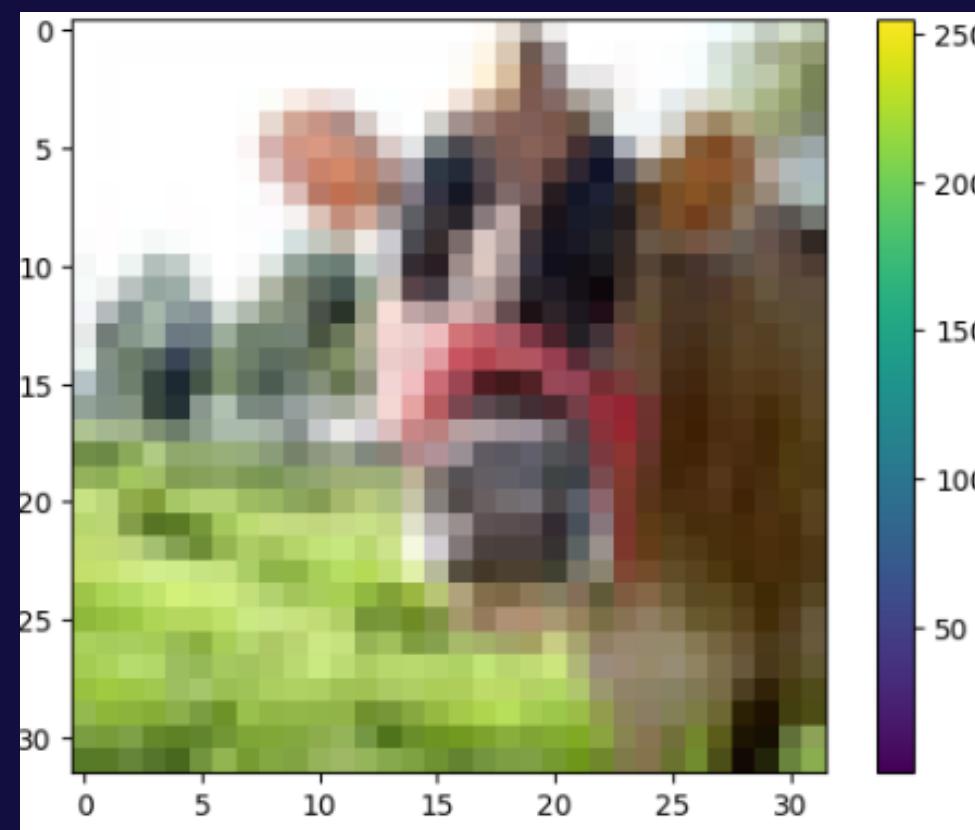
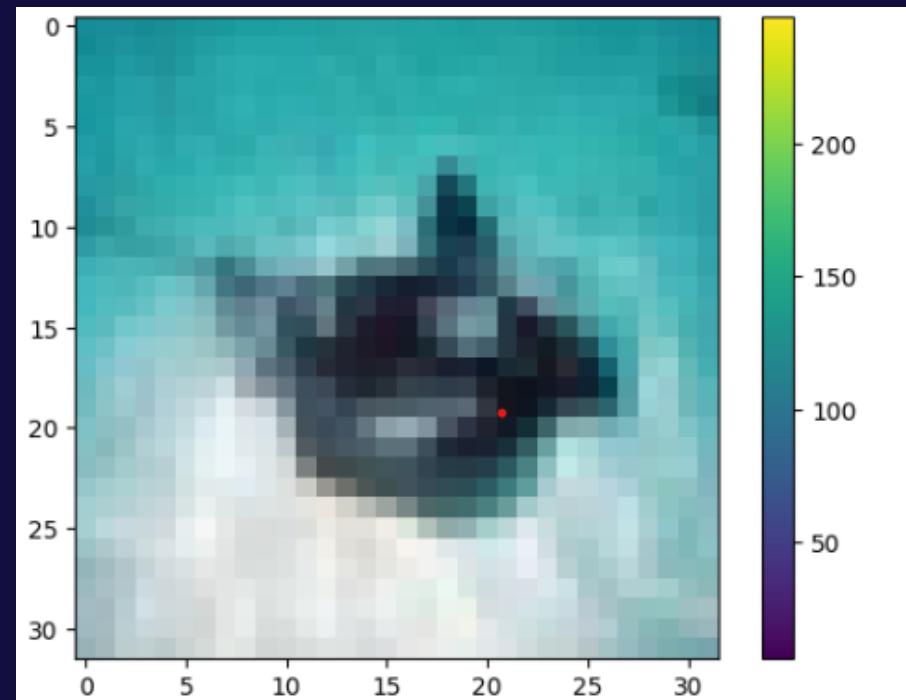
Önemi Yüksek Doğruluk: CNN'ler, genellikle görüntü işleme görevlerinde yüksek doğruluk sağlar. Bu nedenle, sınıflandırma, tanıma ve tespit gibi alanlarda tercih edilirler..Genelleme Yeteneği: CNN'ler, verilerdeki genel desenleri öğrenebilir ve farklı görüntülerde benzer desenleri tanıyalabilir

Model Sonucu:

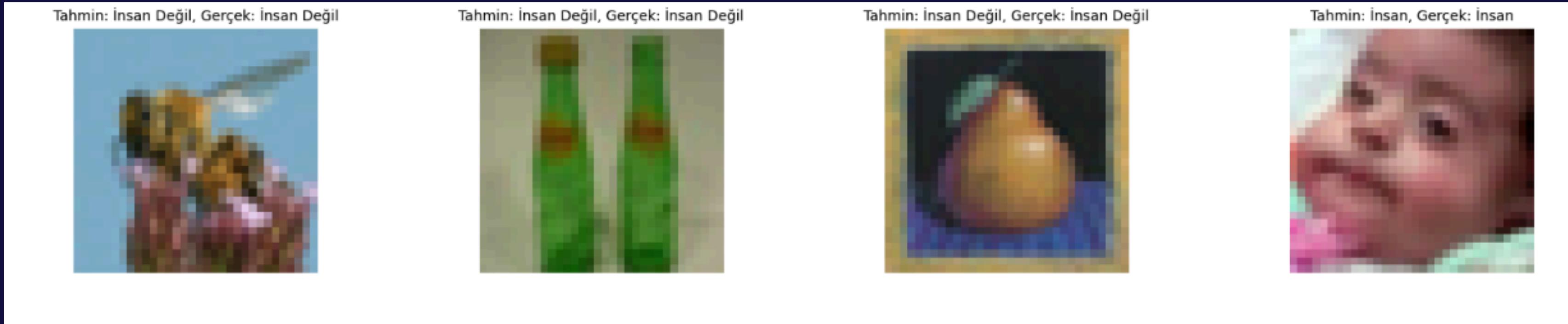
```
Epoch 7/25
782/782 [=====] - 234s 300ms/step - loss: 0.1337 - accuracy: 0.9570 - val_loss: 0.1153 - val_accuracy: 0.9620
Epoch 8/25
782/782 [=====] - 230s 294ms/step - loss: 0.1257 - accuracy: 0.9596 - val_loss: 0.1190 - val_accuracy: 0.9626
Epoch 9/25
782/782 [=====] - 233s 297ms/step - loss: 0.1233 - accuracy: 0.9603 - val_loss: 0.1253 - val_accuracy: 0.9584
Epoch 10/25
782/782 [=====] - 237s 303ms/step - loss: 0.1198 - accuracy: 0.9616 - val_loss: 0.1068 - val_accuracy: 0.9643
Epoch 11/25
782/782 [=====] - 238s 304ms/step - loss: 0.1172 - accuracy: 0.9618 - val_loss: 0.1117 - val_accuracy: 0.9659
Epoch 12/25
782/782 [=====] - 233s 297ms/step - loss: 0.1119 - accuracy: 0.9636 - val_loss: 0.1157 - val_accuracy: 0.9635
Epoch 13/25
782/782 [=====] - 238s 304ms/step - loss: 0.1108 - accuracy: 0.9638 - val_loss: 0.1196 - val_accuracy: 0.9634
Epoch 14/25
782/782 [=====] - 234s 300ms/step - loss: 0.1092 - accuracy: 0.9648 - val_loss: 0.1289 - val_accuracy: 0.9588
Epoch 15/25
782/782 [=====] - 230s 295ms/step - loss: 0.1039 - accuracy: 0.9666 - val_loss: 0.1266 - val_accuracy: 0.9619
Epoch 16/25
782/782 [=====] - 234s 299ms/step - loss: 0.0976 - accuracy: 0.9683 - val_loss: 0.0956 - val_accuracy: 0.9690
Epoch 17/25
782/782 [=====] - 231s 296ms/step - loss: 0.0925 - accuracy: 0.9692 - val_loss: 0.0998 - val_accuracy: 0.9678
Epoch 18/25
782/782 [=====] - 243s 310ms/step - loss: 0.0907 - accuracy: 0.9701 - val_loss: 0.0946 - val_accuracy: 0.9712
Epoch 19/25
782/782 [=====] - 236s 302ms/step - loss: 0.0889 - accuracy: 0.9712 - val_loss: 0.0971 - val_accuracy: 0.9707
Epoch 20/25
782/782 [=====] - 231s 296ms/step - loss: 0.0861 - accuracy: 0.9720 - val_loss: 0.1447 - val_accuracy: 0.9642
Epoch 21/25
782/782 [=====] - 237s 303ms/step - loss: 0.0862 - accuracy: 0.9714 - val_loss: 0.1044 - val_accuracy: 0.9680
Epoch 22/25
782/782 [=====] - 234s 299ms/step - loss: 0.0846 - accuracy: 0.9718 - val_loss: 0.0889 - val_accuracy: 0.9714
Epoch 23/25
782/782 [=====] - 235s 301ms/step - loss: 0.0838 - accuracy: 0.9719 - val_loss: 0.0976 - val_accuracy: 0.9700
Epoch 24/25
782/782 [=====] - 228s 291ms/step - loss: 0.0824 - accuracy: 0.9718 - val_loss: 0.0953 - val_accuracy: 0.9713
Epoch 25/25
782/782 [=====] - 236s 301ms/step - loss: 0.0799 - accuracy: 0.9730 - val_loss: 0.0926 - val_accuracy: 0.9722
```



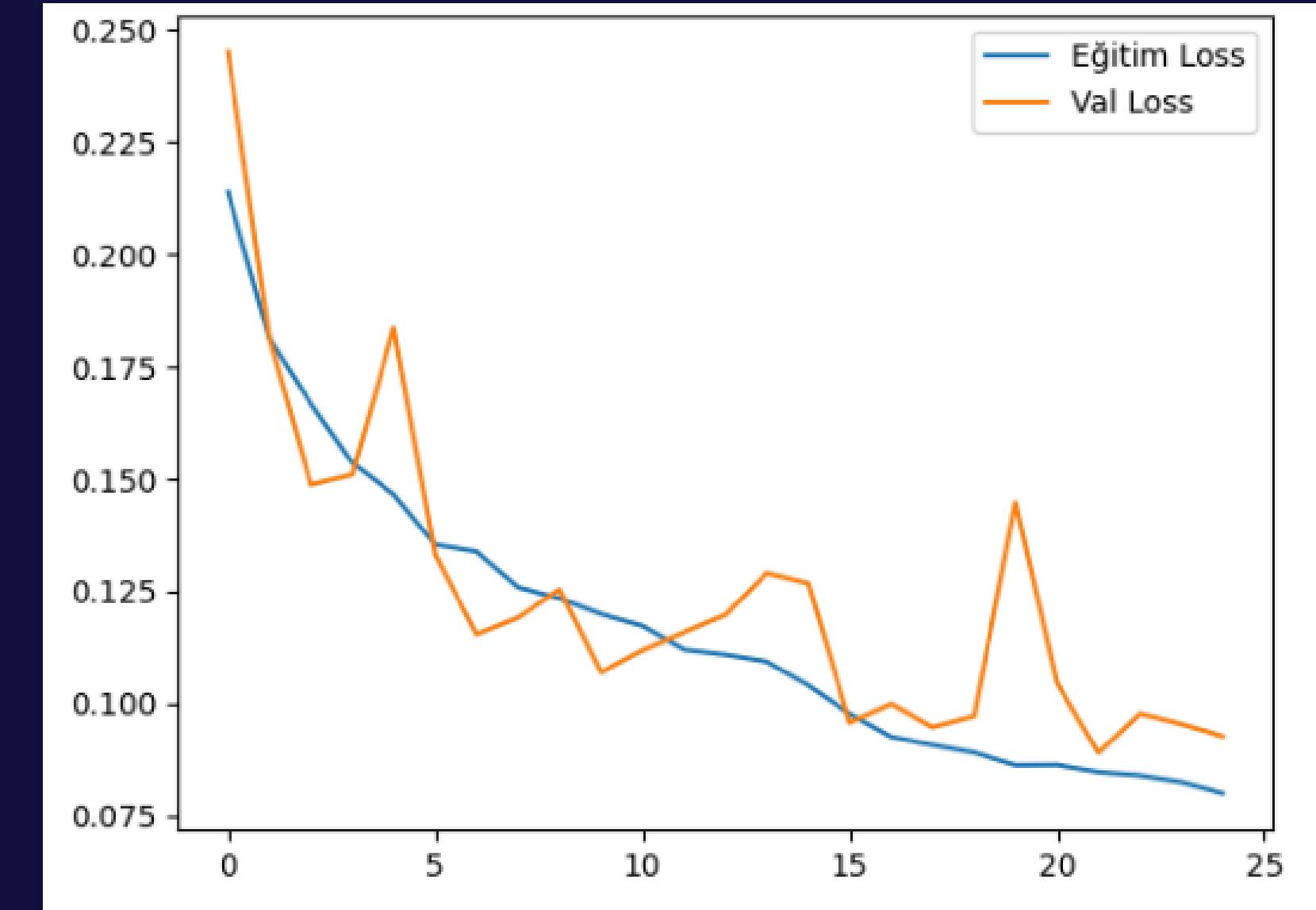
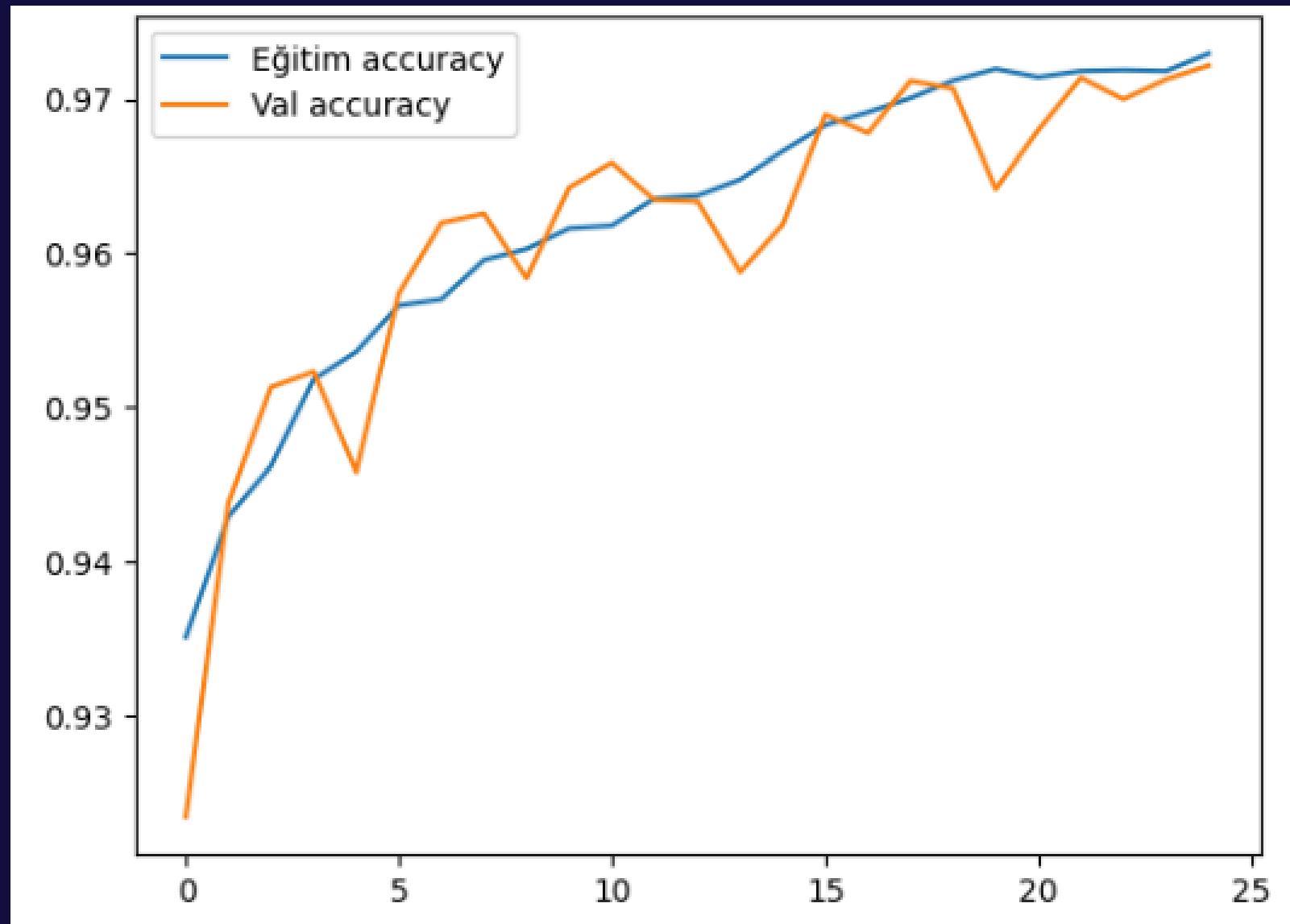
Veri Görselleri



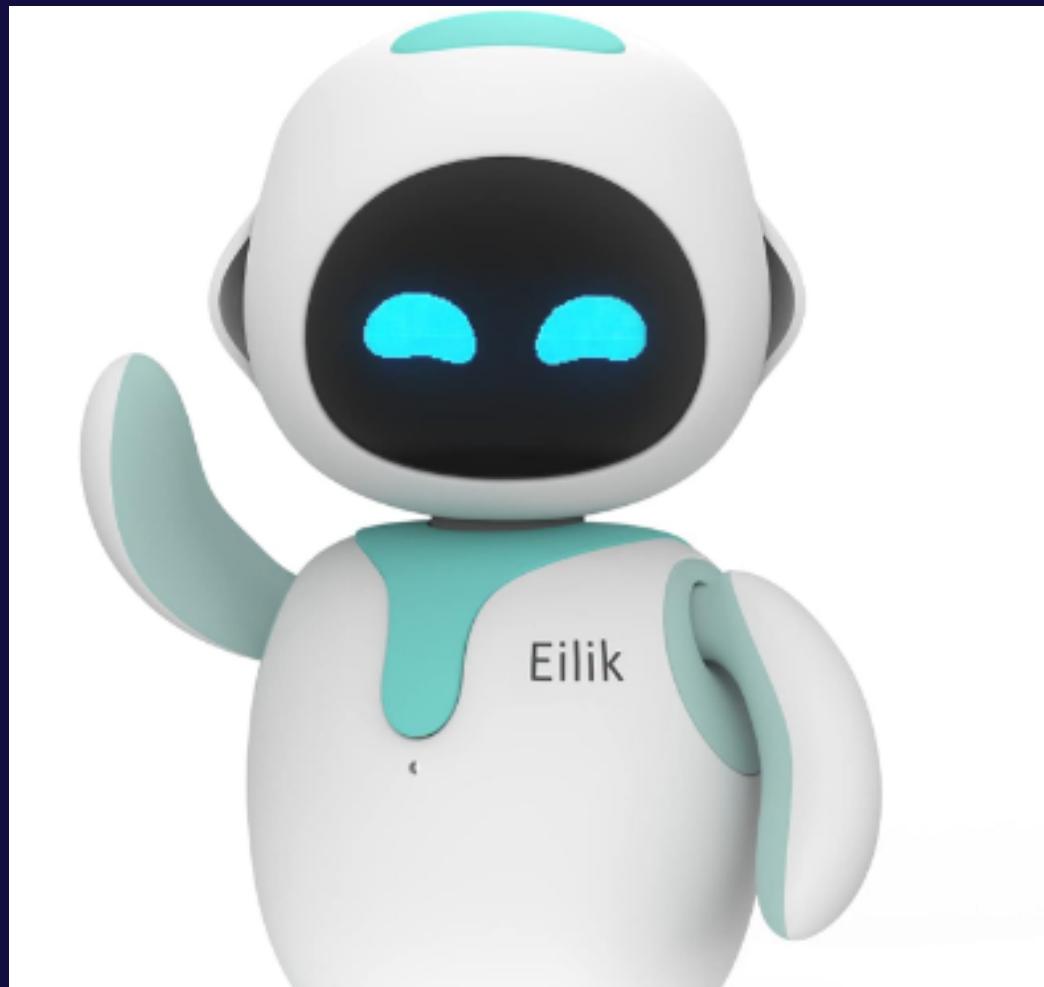
Eğitim Sonrası Görüşeller



Grafikler



Örnek Uygulama



Kullanıcı Fotoğraf Yükleme Butonu

```
[9] from google.colab import files  
img= files.upload()
```

 Dosyaları Seç WhatsApp I...45.38.jpeg
• WhatsApp Image 2024-05-02 at 11.45.38.jpeg(image/jpeg) - 62795 bytes, last modified: 02.05.2024 - 100% done
Saving WhatsApp Image 2024-05-02 at 11.45.38.jpeg to WhatsApp Image 2024-05-02 at 11.45.38.jpeg

Örnek Uygulama Sonucu

```
#Resmin Yüklenmesi:  
img = Image.open(img_path)  
# Resmin yeniden boyutlandırılması:  
resized_img = img.resize((32, 32))  
# Yeniden boyutlandırılmış resmi kaydedilmesi:  
resized_img.save('./photo.jpeg')  
img_path = './photo.jpeg'  
  
img = image.load_img(img_path, target_size=(32, 32))  
img_array = image.img_to_array(img)  
img_array = np.expand_dims(img_array, axis=0)  
img_array = preprocess_input(img_array / 255.0)  
  
prediction = model.predict(img_array)  
print(prediction)  
if prediction >= 0.5:  
    print("Bu bir insan.")  
else:  
    print("Uygun değil, başka bir görselle deneyiniz.")  
predict_image(model,"WhatsApp Image 2024-05-02 at 11.45.38.jpeg")  
  
→ 1/1 [=====] - 0s 325ms/step  
[[0.]]  
Uygun değil, başka bir görselle deneyiniz.
```



Geliştirilmesi Gerekenler :

Projenin diğer adımları için farklı veri setleri ile çalışmak ve bu veri seti ile entegre hale getirmek gerekmektedir.

İnsan görsellerinin net şekilde gözükmesi gerekmektedir aksi halde tahminleme yapamamaktadır.

Veriler 2 sınıfa bölündü ancak bu da veri dengesizliği oluşturdu bu konuda geliştirilmeli.

Başka Ne Yapılabilirdi?

Doğalgaz çıkan bölgelerin yapısını, koordinatını, görüntüsünü vb. kaydederek veri seti haline getirmek ve görüntü işleme ile drone ya da uydu maps vb. görüntüleri ile olası doğalgaz çıkacak yerlerin tespitini yapılması gerçekleştirilebilir.



Teşekkürler

Gizem KARATAŞ  [Linkedin](#)