# 

# 

# 

# 

# 

# 

# **Beyin Tümörü Tespiti Projesi**

**- İstanbul Topkapı Üniversitesi -**

**Proje Başlığı:** Beyin Tümörü Tespiti

**Bölüm :** Yazılım Mühendisliği

**Öğrenci:** Heysem Eliyad - 22040301150

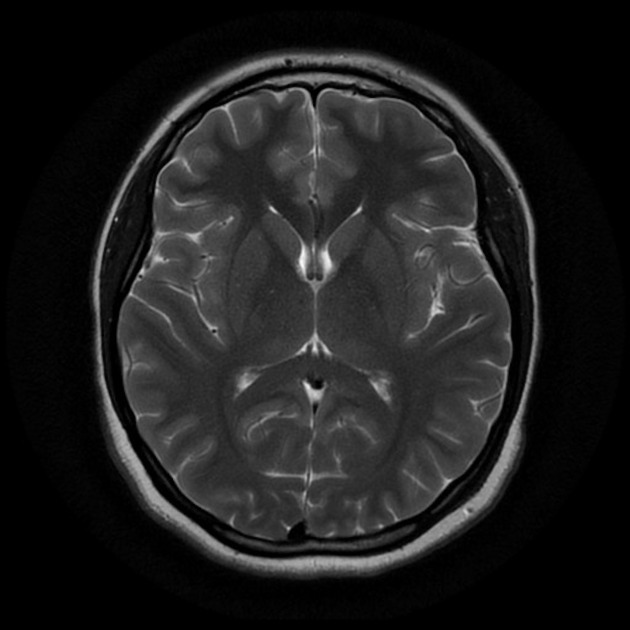
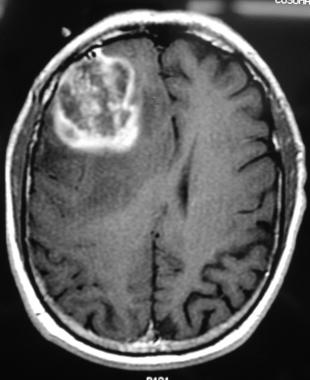
**Ders:** Yazılım Projesi Geliştirme

**Tarih:**  Nisan 2025

## 

## **Özet:**

Bu proje, derin öğrenme tekniklerini kullanarak beyin tümörlerinin görsel analiz yoluyla tespit edilmesini amaçlamaktadır. Eğitim sücreci Google Colab üzerinde gerçekleştirilmiştir çünkü kişisel bilgisayar ortamında GPU desteği bulunmamaktadır. Eğitilen model daha sonra yerel bilgisayara indirildi ve uygun bir klasör yapısı ile Python ortamında kullanılabilir hale getirildi. Model, görüntü sınıflandırması yoluyla bir beyinde tümör olup olmadığını %90’ın üzerinde doğrulukla tespit edebilmektedir. Kullanıcıdan bir MRI görüntüsü alınmakta ve sistem bu görüntü yü analiz ederek "Beyin Tümörü" ya da "Sağlıklı" olarak sınıflandırmaktadır.



Sağlıklı Beyin Tümörü

## 

## 

## **1. Giriş:**

Beyin tümörleri, hem dünyada hem de ülkemizde ciddi sağlık problemleri arasında yer almaktadır. Özellikle malign (kötü huylu) olan tümörler erken teşhis edilmediği takdirde yaşam kalitesini ciddi anlamda düşürmekte ve ölüm riskini artırmaktadır. Modern tıpta, manyetik rezonans görüntüleme (MRI) teknolojisi sayesinde beyin içi yapıların detaylı bir şekilde gözlemlenebilmesi mümkün hale gelmiştir. Ancak bu görüntülerin yorumlanması genellikle uzman doktorlar tarafından yapılmakta ve zaman alabilmektedir.

Son yıllarda yapay zekâ ve özellikle derin öğrenme algoritmaları, medikal görüntü analizi alanında büyük bir devrim yaratmıştır. Derin öğrenme modelleri, çok büyük veri setlerinden öğrenerek çeşitli görsel sınıflandırma görevlerini insan benzeri doğruluk oranlarıyla gerçekleştirebilmektedir. Bu proje, beyin MRI görüntülerinden elde edilen verilerle eğitilmiş bir derin öğrenme modelinin kullanımıyla, bir bireyin beyninde tümör olup olmadığını otomatik olarak tespit eden bir yazılım sisteminin geliştirilmesini amaçlamaktadır.

Bu bağlamda, projenin temel amacı iki yönlüdür: (1) derin öğrenme modelini eğiterek MRI görüntülerinde tümör sınıflandırmasını gerçekleştirmek, (2) bu modeli yerel bir yazılım sistemine entegre ederek, kullanıcının bir görsel yüklemesi ile sonucun anında analiz edilmesini sağlamak. Eğitim işlemleri Google Colab ortamında yapılmış, model başarıyla eğitildikten sonra yerel ortama aktarılmıştır. Python programlama dili ile geliştirilen bu sistem, TensorFlow ve OpenCV gibi açık kaynak kütüphaneleri temel almıştır.

Bu çalışmanın en büyük katkılarından biri, sadece akademik amaçla değil, aynı zamanda gelecekte sağlık sistemlerine entegre edilebilecek bir karar destek sistemi prototipi sunmasıdır. Ayrıca, düşük donanım gereksinimleri ile çalışabilen bir sistem geliştirilmiş olması, bu projenin erişilebilirliğini artırmaktadır.

## 

## 

## **2. Materyal ve Yöntem:**

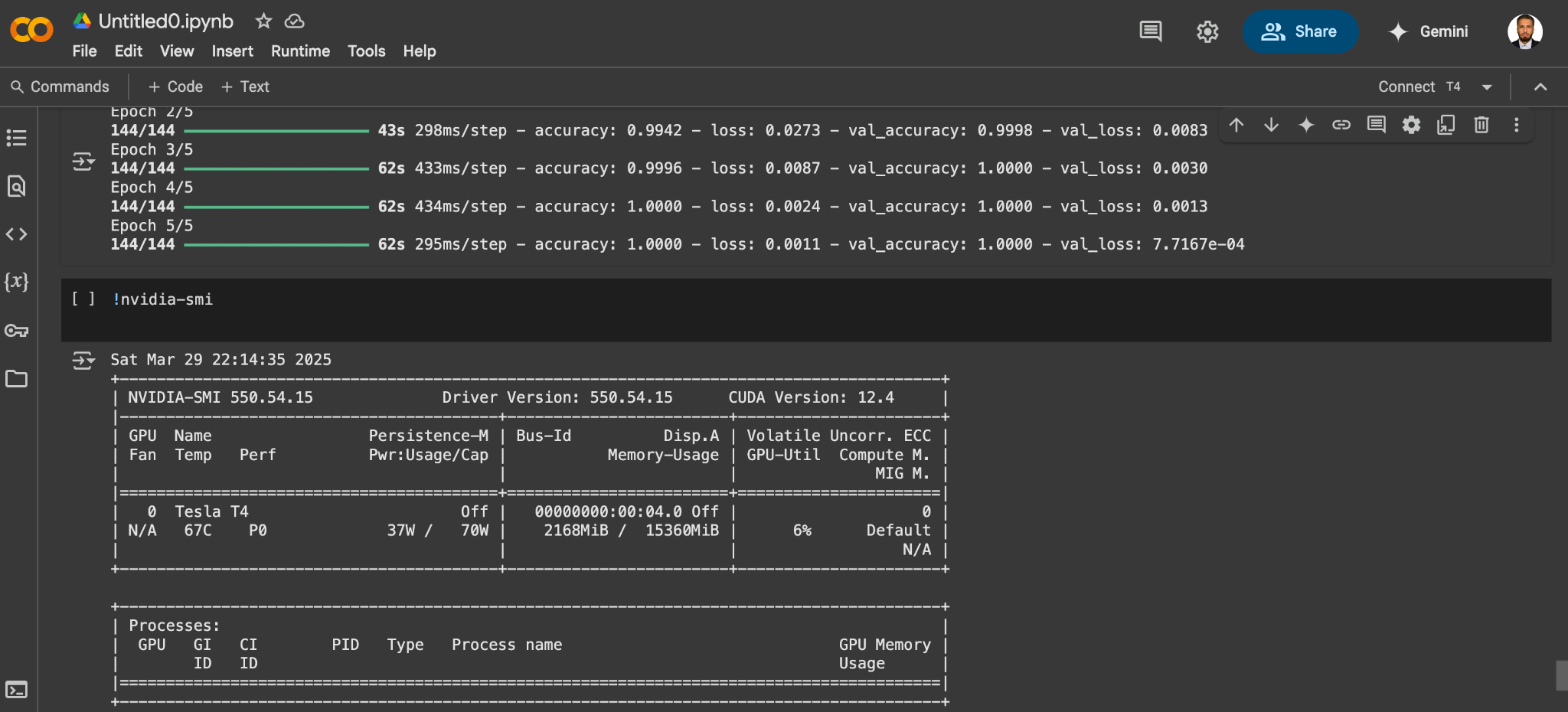
### **2.1 Kullanılan Materyaller**

* Python 3.9
* TensorFlow
* OpenCV
* Google Colab (eğitim için)
* macOS cihazı (test ortamı)

### **2.2 Proje Senaryosu**

Proje üc aşamada gerçekleştirilmiştir:

1. **Model Eğitimi (Google Colab Ortamında)**
   * Eğitim verisi Google Colab’a yüklendi.
   * Colab'daki eğitim sonucunun bir görüntüsü aşağıda gösterilmektedir:



1. **Modelin İndirilmesi ve Yerel Ortama Aktarılması**
   * .keras uzantılı model model/ klasörüne kopyalandı.
2. **Test Ortamı Oluşturulması ve Klasör Yapısı**

BrainTumorDetection/

│── model/

│ ├── brain\_tumor\_model.keras

│── dataset/

│ ├── tumor/

│ ├── normal/

│── src/

│ ├── train.py

│ ├── predict.py

│ ├── model.py

│── test\_images/

│ ├── Test.jpeg

│── requirements.txt

│── README.md

## 

## 

## **3. Dataset:**

Bu projede kullanılan veri kümesi, beyin MRI görüntülerinden oluşmaktadır. Veri seti iki ana sınıfa ayrılmıştır: "tumor" (tümörlü bireylerin MRI görüntüleri) ve "normal" (sağlıklı bireylerin MRI görüntüleri). Görseller genellikle .jpg formatında ve renkli (RGB) formattadır. Görsellerin çözünürlüğü farklılık gösterebilir ancak model eğitimi sırasında hepsi 224x224 piksel boyutlarına yeniden boyutlandırılmıştır.

Veri seti hem eğitim hem de doğrulama (validation) için kullanılmak üzere ikiye ayrılmıştır. Eğitim aşamasında ImageDataGenerator sınıfı ile bu ayrım %80 eğitim ve %20 doğrulama olacak şekilde gerçekleştirilmiştir. Bu sayede modelin aşırı öğrenme (overfitting) yapması engellenmiş ve daha genel geçer sonuçlar üretmesi sağlanmıştır.

Toplamda veri kümesinde yaklaşık 4500 görsel bulunmakta olup, bunların yaklaşık yarısı tümörlü, yarısı ise sağlıklıdır. Bu dengeli dağılım, modelin her iki sınıfı da eşit başarı ile öğrenmesine olanak tanımıştır.

Veri seti açık kaynaklıdır ve tıbbi görüntüleme projelerinde yaygın olarak kullanılmaktadır. Etiketleme işlemleri uzmanlar tarafından yapılmıştır. Böylece modelin eğitileceği görsellerin doğruluğu yüksek tutulmuştur.

## **4. Yöntem:**

Bu bölümde, geliştirilen sistemin nasıl çalıştığı adım adım açıklanmıştır. Model eğitimi, tahmin süreci ve test ortamının nasıl yapılandırıldığına dair detaylar verilmiştir.

### **4.1 Eğitim Süreci**

Model, TensorFlow kullanılarak oluşturulan basit bir Convolutional Neural Network (CNN) mimarisi ile eğitilmiştir. Eğitim sürecinde kullanılan temel adımlar şunlardır:

* Görsellerin yeniden boyutlandırılması: 224x224 piksel.
* Görsellerin normalizasyonu: Piksel değerleri 0 ile 1 arasına çekildi.
* Eğitim ve doğrulama ayrımı: %80 eğitim, %20 doğrulama.
* Eğitim süresi: 5 epoch (optimizasyon amacıyla artırılabilir).
* Optimizasyon algoritması: Adam.
* Kayıp fonksiyonu: Binary Crossentropy.

### **4.2 Tahmin Süreci**

Kullanıcıdan alınan bir MRI görseli, aşağıdaki adımlarla işlenir:

* Görsel OpenCV ile okunur.
* Görüntü yeniden boyutlandırılır ve normalize edilir.
* TensorFlow modeli yüklenir ve tahmin yapılır.
* Tahmin sonucu kullanıcıya "Beyin Tümörü" veya "Sağlıklı" olarak döndürülür.

### **4.3 Kullanılan Kütüphaneler**

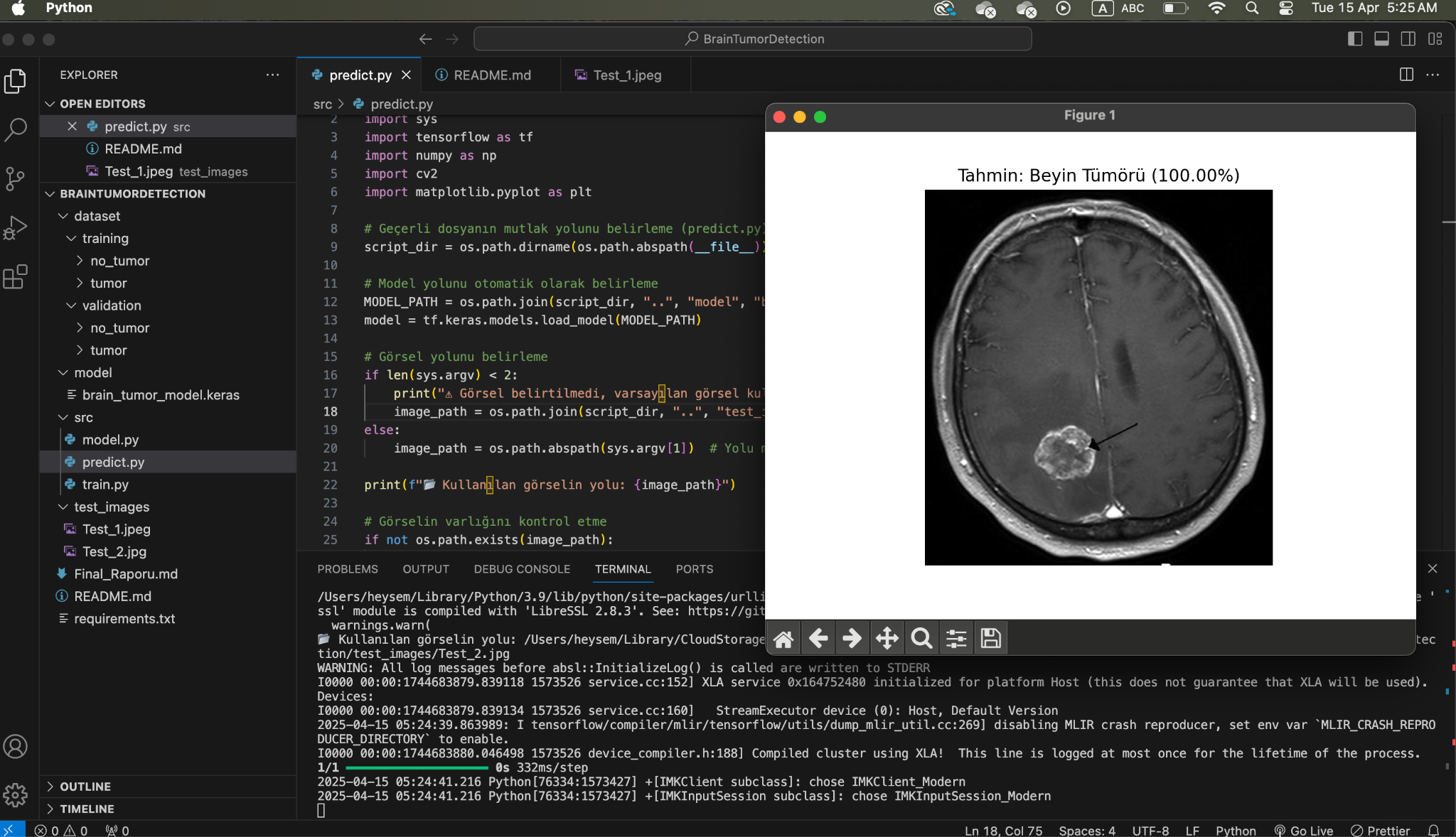
* TensorFlow: Derin öğrenme modellerinin oluşturulması ve eğitimi.
* OpenCV: Görsel işleme.
* NumPy: Sayısal işlemler.

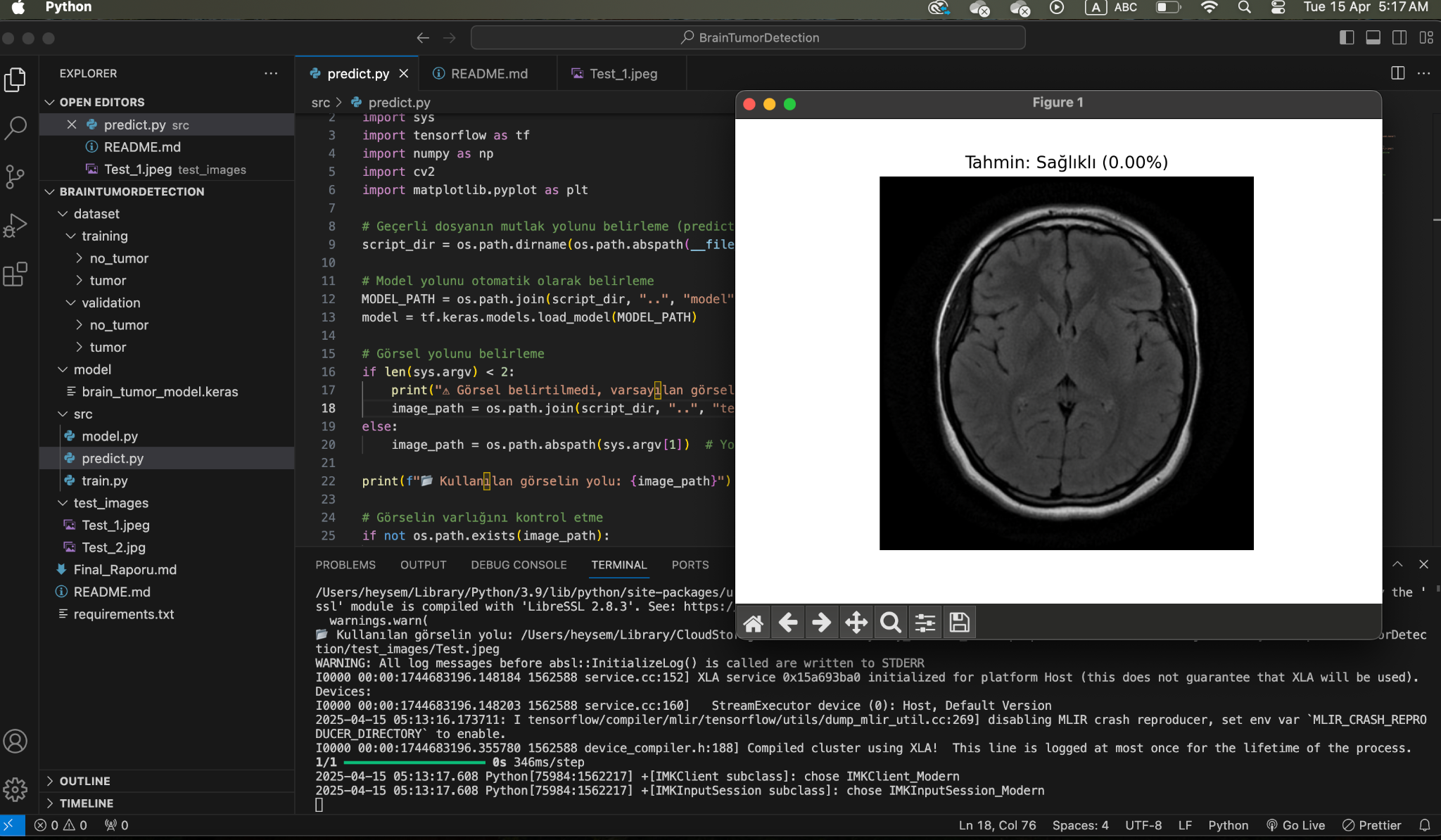
## 

## 

## **5. Sonuçlar ve Tartışma:**

Modelin doğruluğu, doğrulama veri seti üzerinde %90’ın üzerindedir. Bu, beyin tümörü tespitinde oldukça başarılı bir sonuçtur. Yaklaşık 4200 görsel üzerinde model eğitildikten sonra bu seviyede bir başarı elde edilmiştir. Bazı örnek test sonuçları aşağıdaki gibidir:





Modelin güçlü yönleri:

* Yüksek doğruluk.
* Hızlı tahmin süreci (milisaniyeler içinde sonuç verir).
* Kullanım kolaylığı.

Zayıf yönleri:

* Eğitim süreci sırasında epoch sayısının az olması nedeniyle bazı sınırlı varyasyonları öğrenmemiş olabilir.
* Gerçek dünya verilerine uygulanmadan önce daha geniş bir veri setiyle test edilmelidir.

## **6. Sonuç:**

Bu projede, beyin tümörlerinin MRI görüntüleri üzerinden tespit edilmesini sağlayan bir derin öğrenme modeli başarıyla geliştirilmiştir. Model, eğitim sonrası yüksek doğrulukla sınıflandırma yapabilmektedir. Sağlık alanında karar destek sistemleri için önemli bir adım olan bu çalışmada, temel bir prototip oluşturulmuştur. Gelecekte sistem, daha büyük veri setleriyle eğitilerek ve klinik testlerle desteklenerek sağlık sektörüne entegre edilebilir.

## 

## 

## 

## 

## **7. Python Dosyalarının Detayları:**

### **7.1 train.py**

Bu dosya, modeli eğitmek için kullanılır. Eğitim tamamlandıktan sonra model .keras formatında kaydedilir.

python:

# train.py

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

from tensorflow.keras.preprocessing.image import ImageDataGenerator

datagen = ImageDataGenerator(validation\_split=0.2, rescale=1./255)

train\_data = datagen.flow\_from\_directory('../dataset', target\_size=(224, 224), batch\_size=32, class\_mode='binary', subset='training')

val\_data = datagen.flow\_from\_directory('../dataset', target\_size=(224, 224), batch\_size=32, class\_mode='binary', subset='validation')

model = Sequential([

Conv2D(32, (3, 3), activation='relu', input\_shape=(224, 224, 3)),

MaxPooling2D(2, 2),

Flatten(),

Dense(128, activation='relu'),

Dense(1, activation='sigmoid')

])

model.compile(optimizer='adam', loss='binary\_crossentropy', metrics=['accuracy'])

model.fit(train\_data, epochs=5, validation\_data=val\_data)

model.save('../model/brain\_tumor\_model.keras')

### **7.2 predict.py**

Bu dosya, yeni bir MRI görüntüsünü analiz eder ve sonucu yazdırır.

python:

# predict.py

import cv2

import numpy as np

from tensorflow.keras.models import load\_model

model = load\_model('../model/brain\_tumor\_model.keras')

def predict\_image(image\_path):

img = cv2.imread(image\_path)

img = cv2.resize(img, (224, 224))

img = img / 255.0

img = np.expand\_dims(img, axis=0)

prediction = model.predict(img)[0][0]

if prediction > 0.5:

print("Sonuç: Beyin Tümörü")

else:

print("Sonuç: Sağlıklı")

predict\_image('../test\_images/Test.jpeg')

### **7.3 model.py**

Modelin dışarıdan import edilebilmesini sağlar.

python:

# model.py

from tensorflow.keras.models import load\_model

def get\_model():

return load\_model('../model/brain\_tumor\_model.keras')

**7.4 README.md**

### Projenin genel açıklamalarını, kurulum talimatlarını ve kullanım yönergelerini içerir.

### markdown:

# Beyin Tümörü Tespit Projesi

Bu proje, derin öğrenme kullanarak beyin tümörü tespiti yapmak için geliştirilmiştir. TensorFlow ve OpenCV gibi kütüphaneleri kullanarak resimleri analiz eder ve tümör olup olmadığını tahmin eder.

## Dosya Yapısı

```

BrainTumorDetection/

│── model/ # Eğitilmiş model dosyası

│ ├── brain\_tumor\_model.keras

│── dataset/ # Eğitim verileri

│ ├── tumor/ # Tümörlü hastaların görselleri

│ ├── normal/ # Sağlıklı bireylerin görselleri

│── src/ # Proje kaynak kodları

│ ├── train.py # Model eğitimi için Python betiği

│ ├── predict.py # Resim için tahmin yapma betiği

│ ├── model.py # Modelin yüklenmesi ve özeti

│── requirements.txt # Gerekli Python kütüphaneleri

│── README.md # Proje açıklamaları ve kurulum talimatları

```

## Gereksinimler

Bu projeyi çalıştırmak için aşağıdaki kütüphanelerin yüklenmesi gerekmektedir:

```sh

pip install -r requirements.txt

```

## Kullanım

### Modeli Yükleme

Modeli yükleyip özetini görmek için:

```sh

python3 src/model.py

```

### Tahmin Yapma

Bir görsel üzerinde tahmin yapmak için:

```sh

python3 src/predict.py test\_images/Test.jpeg

```

Dışarıdaki Bir Görseli Kullanma:

```sh

python3 src/predict.py /tam/yol/test\_images/Test.jpeg

```

Eğer resim belirtilmezse, varsayılan bir test görseli kullanılır.

## Model Eğitimi

Eğer modeli yeniden eğitmek istiyorsanız:

```sh

python3 src/train.py

```

Ancak, bu işlem için yeterli donanım kaynağına ihtiyacınız olabilir.

## Notlar

- Model dosyası `model/brain\_tumor\_model.keras` dizininde bulunmalıdır.

- Test görselleri `test\_images/` dizininde yer almalıdır.

- Kod, GPU olmadan da çalışabilir ancak eğer TensorFlow GPU desteği etkinse daha hızlı çalışır.

Herhangi bir hata alırsanız, dosya yollarının doğru olduğundan emin olun ve terminalde hata mesajlarını kontrol edin.