Uygulama 9. Tekrarlı Sinir Ağı (RNN) Derin Öğrenme Modelinin Uygulanması

Bu uygulama ile diyabetli hastaların RNN ile analizi gerçekleştirilecektir. Diyabet verisetine ait değerler kullanılarak iki sınıflı bir sınıflandırma işlemi gerçekleştirilecektir.

Öncelikle gerekli kütüphaneler yüklenir.

```
#kütüphanelerin yüklenmesi
import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder
```

Ardından veriseti yüklenir ve verideki sınıflar etiketlenir.

```
#verisetinin yüklenmesi
veri = pd.read_csv("C:/Users/burak/Masaüstü/YMT410 Yapay Zeka ve Uzman Sistemler/Uygulama #9/diyabet.csv")

#Sınıf sayısı ve etiketinin belirlenmesi
label_encoder = LabelEncoder().fit(veri.output)
labels = label_encoder.transform(veri.output)
classes = list(label_encoder.classes_)
```

Girdi ve çıktı verileri hazırlanır, standartlaştırma işlemi yapılır.

```
#Girdi ve çıktı verilerinin hazırlanması
X = veri.drop(["output"],axis=1)
y = labels
nb_features = 8
nb_classes = len(classes)


#verilerin standartlaştırılması
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X = sc.fit_transform(X)
```

Eğitim ve test verileri hazırlanarak kategorileştirme işlemi gerçekleştirilir.

```
from sklearn.model_selection import train_test_split
X_train, X_test,  y_train, y_test = train_test_split(X,y,test_size = 0.3)

#çıktı değerlerinin kategorileştirilmesi
from tensorflow.keras.utils import to_categorical
y_train = to_categorical(y_train)
y_test = to_categorical(y_test)
```

Ardından girdiler RNN yapısına uygun olacak şekilde yeniden boyutlandırılır.

```
#Girdi verilerinin yeniden boyutlandırılması
X_train = np.array(X_train).reshape(537,8,1)
X_test = np.array(X_test).reshape(231,8,1)
```

Ardından ağın modeli oluşturulur.

```
#Modelin oluşturulması
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Activation, Dropout
from tensorflow.keras.layers import Flatten, SimpleRNN, BatchNormalization

model = Sequential()
model.add(SimpleRNN(512,input_shape=(nb_features,1)))
model.add(Activation("relu"))
model.add(Dropout(0.25))
model.add(BatchNormalization())
model.add((Flatten()))
model.add(Dense(2048, activation="relu"))
model.add(Dense(1024, activation="relu"))
model.add(Dense(nb_classes, activation="sigmoid"))
model.summary()
```

Daha sonra model derlenir ve eğitilir.

```
#modelin derlenmesi
#optimizasyonun belirlenmesi
from tensorflow.keras.optimizers import SGD
opt = SGD(lr=1e-3, decay=1e-5, momentum=0.9, nesterov=True)

model.compile(loss="binary_crossentropy", optimizer = opt, metrics=["accuracy"])

#modelin eğitilmesi
score = model.fit(X_train, y_train, epochs = 250, validation_data=(X_test,y_test))
```

Ardından ortalama başarımlar ve değerler belirlenir.

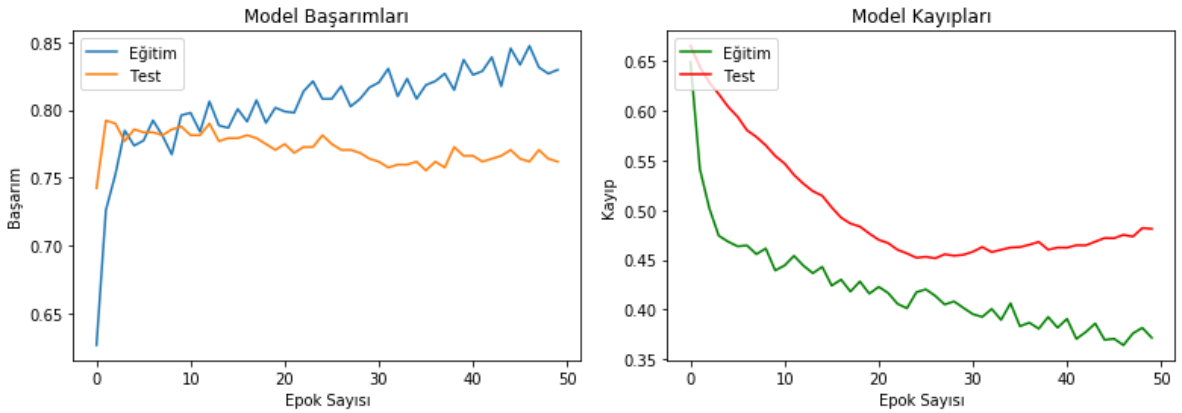
```
#Bilgiler
print(("Ortalama Eğitim Kaybı: ", np.mean(model.history.history["loss"])))
print(("Ortalama Eğitim Başarımı: ", np.mean(model.history.history["accuracy"])))
print(("Ortalama Doğrulama Kaybı: ", np.mean(model.history.history["val_loss"])))
print(("Ortalama Doğrulama Başarımı: ", np.mean(model.history.history["val_accuracy"])))
```

Daha sonra sonuçlar grafik üzerinde gösterilir.

```
#Sonuçların gösterilmesi
import matplotlib.pyplot as plt
plt.plot(model.history.history["accuracy"])
plt.plot(model.history.history["val_accuracy"])
plt.title("Model Başarımları")
plt.ylabel("Başarım")
plt.xlabel("Epok Sayısı")
plt.legend(["Eğitim","Test"], loc="upper left")
plt.show()

plt.plot(model.history.history["loss"],color="g")
plt.plot(model.history.history["val_loss"],color="r")
plt.title("Model Kayıpları")
plt.ylabel("Kayıp")
plt.xlabel("Epok Sayısı")
plt.legend(["Eğitim","Test"], loc="upper left")
plt.show()
```

Örnek çıktı



Uygulama 9. Değerlendirme Soruları

1) 4. Haftada verilen "telefon_fiyat_değişimi.csv" verisini RNN modelini oluşturarak analiz ediniz. Analiz işleminiz sonunda kayıp ve başarım değerlerini çıktı olarak veriniz. Bunlara ek olarak sonuçları grafik üzerinde gösteriniz **(25p).**

2) 5. Haftada verilen yaprak verisetini RNN modelini oluşturarak analiz ediniz. Analiz işleminiz sonunda kayıp ve başarım değerlerini çıktı olarak veriniz. Bunlara ek olarak sonuçları grafik üzerinde gösteriniz **(25p).**

3) Kaktüs bitkisine ait verileri kullanarak RNN derin öğrenme modeli ile sınıflandırma gerçekleştiriniz. Sınıflandırma sonuçlarını başarım, f1-skor, hassaslık, duyarlılık ve AUC skorlarını kullanarak değerlendiriniz **(50p)**. Verisetine https://www.kaggle.com/c/aerial-cactus-identification/data adresinden ulaşılabilir.