T.C.

FIRAT ÜNİVERSİTESİ

TEKNOLOJİ FAKÜLTESİ

YAZILIM MÜHENDISLIĞI YMH 418 GÜNCEL KONULAR DERSİ PROJE DOSYASI

01.06.2020 – 05.06.2020

BÖLÜMÜ : YAZILIM MÜHENDİSLİĞİ

NUMARASI : 15542525

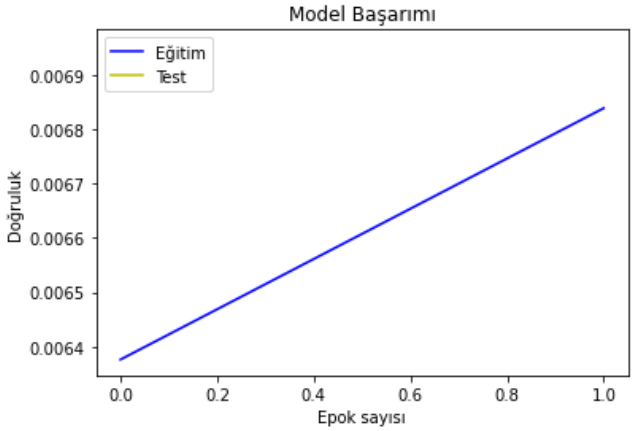
ADI ve SOYADI : ÇAĞDAŞ KARACA

IV. Aşama - Model Kurma:

Projede yapay zeka dersinde edinilen lstm ile sınıflandırma işlemi proje üzerinde uygulanmıştır. Veri seti üzerinde okunan değerlerden state Date ve State değişkenlerinin veri setinden kaldırılıp okunması sağlanmıştır. Model üzerinde f1 skor başarım ve doğruluk değerleri hesaplanmıştır. Bu işlemin kodu şu şekildedir.

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Activation,LSTM,BatchNormalization,Dropout,Flatten
from keras import backend as K

veri = pd.read_csv('C:/Users/cagda/OneDrive/Masaüstü/Çağdaş Karaca Hava
Kirliliği/yapaysiniragi/data/i_Detaylar19617703.04.2020_18_19_52.csv')
label_encoder=LabelEncoder().fit(veri.Date)
labels=label_encoder.transform(veri.Date)
classes=list(label_encoder.classes_)
veri=veri.drop(["Date","state"],axis=1)

nb_classes=len(classes)

scaler= StandardScaler().fit(veri.values)
veri=scaler.transform(veri.values)

X_train,X_valid,y_train,y_valid=train_test_split(veri,labels,test_size=0.2)

y_train=to_categorical(y_train)
y_valid=to_categorical(y_valid)

X_train=np.array(X_train).reshape(20703, 4,1)
X_valid=np.array(X_valid).reshape(5176, 4,1)

model=Sequential();

model.add(LSTM(512,input_shape=(4,1)))
model.add(Activation("relu"))
model.add(BatchNormalization())
model.add(Flatten())
model.add(Dropout(0.25))
model.add(Dense(2048,activation="relu"))
model.add(Dense(1024,activation="relu"))
model.add(Dense(nb_classes,activation="softmax"))
model.summary()

def recall_m(y_true,y_pred):
    true_positives=K.sum(K.round(K.clip(y_true*y_pred,0,1)))
    possible_positive=K.sum(K.round(K.clip(y_true,0,1)))
    recall=true_positives/(possible_positive+K.epsilon())
    return recall
def precision_m(y_true,y_pred):
    true_positives=K.sum(K.round(K.clip(y_true*y_pred,0,1)))
```

```
    predicted_positives=K.sum(K.round(K.clip(y_pred,0,1)))
    precision=true_positives/(predicted_positives+K.epsilon())
    return precision

def f1_m(y_true,y_pred):
    precision=precision_m(y_true, y_pred)
    recall=recall_m(y_true, y_pred)
    return 2*((precision*recall)/(precision+recall+K.epsilon()))

model.compile(loss="categorical_crossentropy",optimizer="adam",metrics=["accuracy",f1_m,precision_m,
recall_m])

score=model.fit(X_train,y_train,epochs=15,validation_data=(X_valid,y_valid))

print(("Ortalama Eğitim Başarısı",np.mean(model.history.history["accuracy"])))
print(("Ortalama Doğrulama Başarısı",np.mean(model.history.history["val_accuracy"])))

import matplotlib.pyplot as plt
plt.plot(model.history.history['accuracy'],color="b")
plt.plot(model.history.history['val_accuracy'],color="y")
plt.title("Model Başarımı")
plt.ylabel("Doğruluk")
plt.xlabel("Epok sayısı")
plt.legend(["Eğitim","Test"],loc="upper left")
plt.show()
```

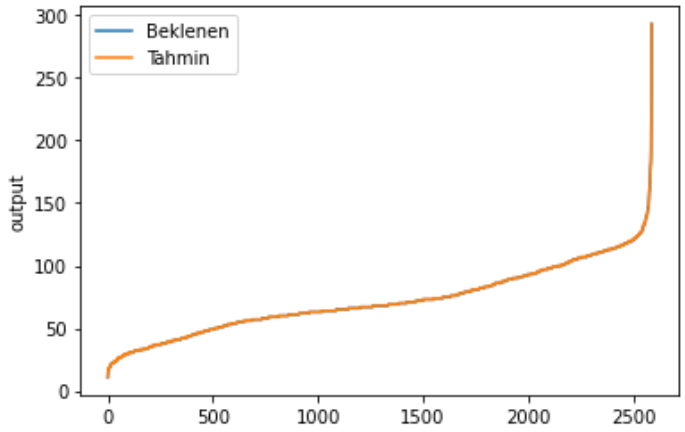


Görsel 1. LSTM ile sınıflandırma işlemi

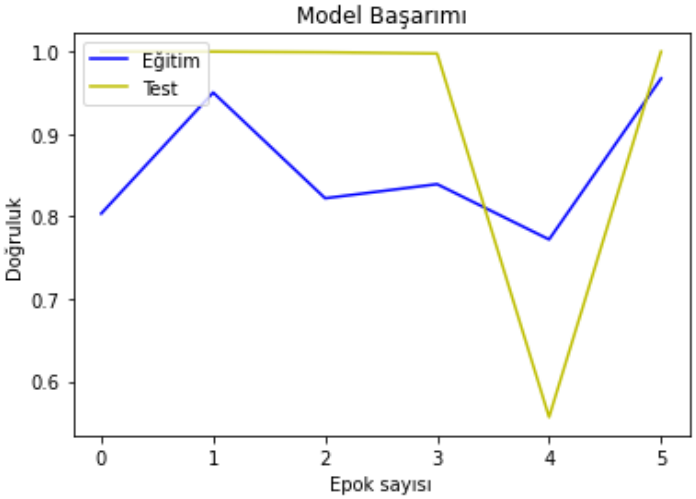
Pythonda oluşturulan yapay sinir ağı ile oluşturularak k means kümeleme algoritması ile veri seti üzerinde yapılan kümeleme işlemi sonuçları şu şekildedir;

```
Kat puanı (RMSE): 0.044751691281048715
Final, örnek dışı skor (RMSE): 0.044751691281048715
('Ortalama Eğitim Başarısı', 0.8591075)
('Ortalama Doğrulama Başarısı', 0.9255894919236501)
('Ortalama 'Eğitim Kaybı', 6.956798865816162)
('Ortalama Doğrulama kaybı', 0.05066158099921711)
```

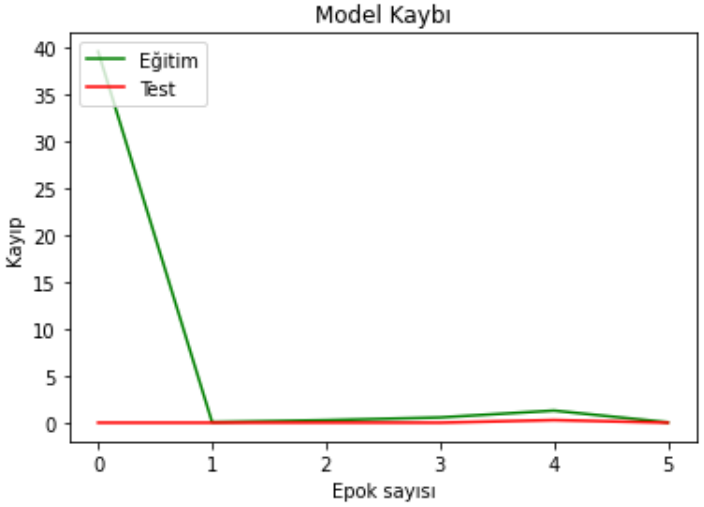
Ortalama eğitim başarımız yapay sinir ağı ile oluşturulan model ile ortalama %85 ile başarılı sonuçlandırılmıştır. Oluşturulan modelin ortalama eğitim kaybı ise %7 dir. Oluşturulan modele göre elde edilen grafik sonuçları aşağıda gösterilmiştir.



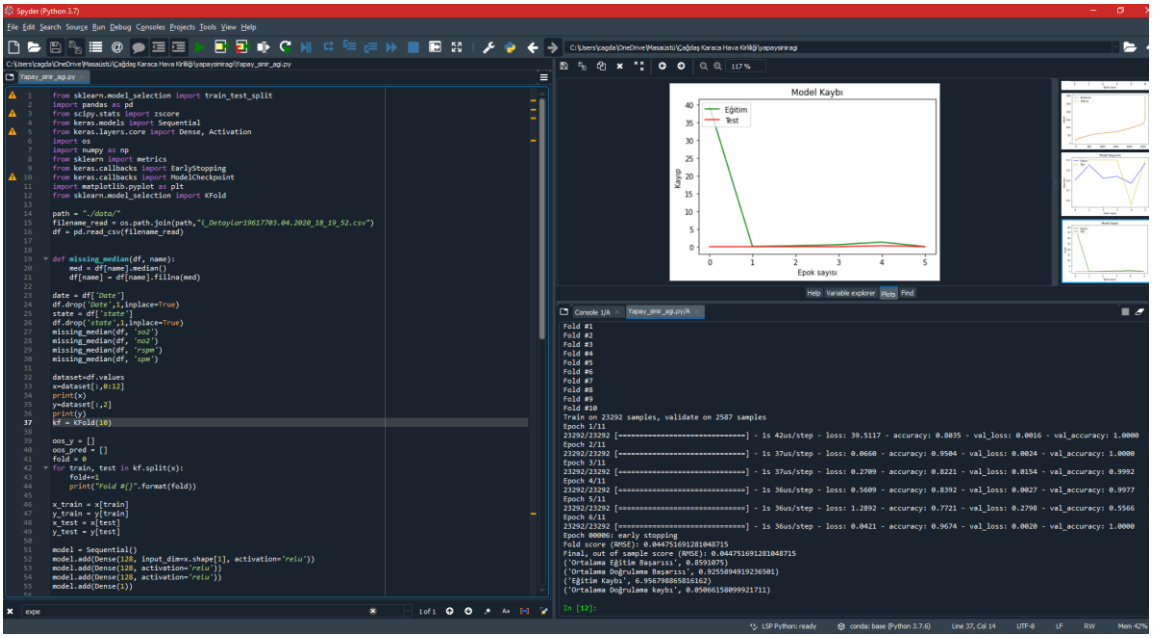
Görsel 2. Kümeleme işleminde tahmin ve beklenen değerlerin görselleştirilmesi



Görsel 3. Modelin eğitim ve test verilerine göre başarım değerlerinin görselleştirilmesi



Görsel 4. Modelin eğitim ve test verilerine göre kayıp değerlerinin görselleştirilmesi



Görsel 5. Sonuçların Ekran Görüntüsü

K means algoritmasının veri seti üzerinde CNN sinir ağları kullanılarak oluşturulan python kodu şu şekildedir:

```
from sklearn.model_selection import train_test_split
import pandas as pd
from scipy.stats import zscore
from keras.models import Sequential
from keras.layers.core import Dense, Activation
import os
import numpy as np
from sklearn import metrics
from keras.callbacks import EarlyStopping
from keras.callbacks import ModelCheckpoint
import matplotlib.pyplot as plt
from sklearn.model_selection import KFold

path = "./data/"
filename_read = os.path.join(path,"i_Detaylar19617703.04.2020_18_19_52.csv")
df = pd.read_csv(filename_read)

def missing_median(df, name):
    med = df[name].median()
    df[name] = df[name].fillna(med)
```

```
date = df['Date']
df.drop('Date',1,inplace=True)
state = df['state']
df.drop('state',1,inplace=True)
missing_median(df, 'so2')
missing_median(df, 'no2')
missing_median(df, 'rspm')
missing_median(df, 'spm')
dataset=df.values
x=dataset[:,0:12]
print(x)
y=dataset[:,2]
print(y)
kf = KFold(10)
oos_y = []
oos_pred = []
fold = 0
for train, test in kf.split(x):
    fold+=1
    print("Fold #{}".format(fold))

x_train = x[train]
y_train = y[train]
x_test = x[test]
y_test = y[test]

model = Sequential()
model.add(Dense(128, input_dim=x.shape[1], activation='relu'))
model.add(Dense(128, activation='relu'))
model.add(Dense(128, activation='relu'))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam', metrics=["accuracy"])
monitor = EarlyStopping(monitor='val_loss', min_delta=1e-3, patience=5, verbose=1, mode='auto')
model.fit(x_train,y_train,validation_data=(x_test,y_test),callbacks=[monitor],verbose=1,epochs=11)
pred = model.predict(x_test)
oos_y.append(y_test)
oos_pred.append(pred)

score = np.sqrt(metrics.mean_squared_error(pred,y_test))
```

```
print("Fold score (RMSE): {}".format(score))
oos_y = np.concatenate(oos_y)
oos_pred = np.concatenate(oos_pred)
score = np.sqrt(metrics.mean_squared_error(oos_pred,oos_y))
print("Final, out of sample score (RMSE): {}".format(score))
def chart_regression(pred, y, sort=True):
    t = pd.DataFrame({'pred': pred, 'y': y.flatten()})
    if sort:
        t.sort_values(by=['y'], inplace=True)
    plt.plot(t['y'].tolist(), label='Beklenen')
    plt.plot(t['pred'].tolist(), label='Tahmin')
    plt.ylabel('output')
    plt.legend()
    plt.show()

chart_regression(pred.flatten(),y_test)
print(("Ortalama Eğitim Başarısı",np.mean(model.history.history["accuracy"])))
print(("Ortalama Doğrulama Başarısı",np.mean(model.history.history["val_accuracy"])))
print(("Eğitim Kaybı",np.mean(model.history.history["loss"])))
print(("Ortalama Doğrulama kaybı",np.mean(model.history.history["val_loss"])))
import matplotlib.pyplot as plt
plt.plot(model.history.history['accuracy'],color="b")
plt.plot(model.history.history['val_accuracy'],color="y")
plt.title("Model Başarımı")
plt.ylabel("Doğruluk")
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 Kaybı")
plt.ylabel("Kayıp")
plt.xlabel("Epok sayısı")
plt.legend(["Eğitim","Test"],loc="upper left")
plt.show()
```

Saygılarımla,

15542525 Çağdaş Karaca