



T.C.
FIRAT ÜNİVERSİTESİ
TEKNOLOJİ FAKÜLTESİ

YAZILIM MÜHENDİSLİĞİ
YAZILIM MÜHENDİSLİĞİNDE
GÜNCEL KONULAR
4.Aşama 2.Rapor

GİZEM ÇOBAN

175541307

PROJE DANIŞMANI

DOÇ. DR. FATİH ÖZKAYNAK

2019-2020

1. Giriş

Bu hafta 4. Aşamada daha önceden belirlemiş olduğumuz veri seti için araştırılan modellerin Python dilinde kodlanıp başarı ve F1-Score değerlerine bakıldı.

2. Yapılan Çalışmalar

Proje uygun olduğunu düşünülen modellerimiz şu şekildedir;

DecisionTreeClassifier

Bir Karar Ağacı, örnekleri sınıflandırmak için basit bir temsildir. Verilerin belirli bir parametreye göre sürekli olarak bölündüğü bir Makine Öğrenimidir. Karar ağacında yani DecisionTreeClassifier algoritmasında öncelikle bir ağaç oluşturulur.

Python'daki kodları Görsel 1'de yer almaktadır.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
"""
Created on Fri May 15 18:11:23 2020

@author: Gizem Çoban
"""

# DecisionTreeClassifier

# Kütüphanelerin import edilmesi
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Veri setinin okunması ve bağımsız-bağımlı değişkenlerin ayrılması
dataset = pd.read_csv('hki.csv')

X=dataset.drop(["Tarih","HKI"], axis=1)
y=dataset["HKI"]
# Feature Scaling (Özellik Ölçekleme)
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X = sc.fit_transform(X)
# Veri setinin eğitim ve test olarak ayrılması
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

# Eğitim veri setine Decision Tree sınıflandırıcısının uygulanması
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
classifier.fit(X_train, y_train)

# Test verileri ile tahmin yapılması
y_pred = classifier.predict(X_test)

# Confusion matrisinin oluşturulması
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

#Accuracy (sınıflandırma için başarımlı metriği) ölçülmesi
from sklearn.metrics import accuracy_score
acc = accuracy_score(y_test, y_pred)
print("Accuracy", acc)

#f1_score (sınıflandırma için başarımlı metriği) ölçülmesi
from sklearn.metrics import f1_score
f1_score = f1_score(y_test, y_pred)
print("f1 score", f1_score)
```

Görsel 1: DecisionTreeClassifier

Çıkan sonuç Görsel 2’de yer almaktadır.

```
In [6]: runfile('C:/Users/ASUS/Desktop/VeriBilimi_eski/
DecisionTreeClassifier.py', wdir='C:/Users/ASUS/Desktop/
VeriBilimi_eski')
Accuracy 1.0
f1 score 1.0
```

Görsel 2: DecisionTreeClassifier Sonucu

Bu algoritma eğitim sonucunda ezber yapmıştır. Bu sebepten dolayı projede kullanılması uygun değildir.

KNeighborsClassifier (KNN)

K En Yakın Komşu (KNN) çok basit, anlaşılması kolay, çok yönlü ve en üst düzey makine öğrenme algoritmalarından biridir. KNN algoritması hem sınıflandırma hem de regresyon problemleri için kullanılır. Özellik benzerliği yaklaşımına dayalı bir algoritmadır. KNN finans, sağlık, siyaset bilimi, el yazısı algılama, görüntü tanıma ve video tanıma gibi çeşitli uygulamalarda kullanılır.

Python’daki kodları Görsel 3’de yer almaktadır.

```
1 | K-Nearest Neighbors (K-NN)
2
3 | # Kütüphanelerin import edilmesi
4 | import numpy as np
5 | import matplotlib.pyplot as plt
6 | import pandas as pd
7
8 | # Veri setinin okunması ve bağımsız-bağımlı değişkenlerin ayrılması
9 | dataset = pd.read_csv('hki.csv')
10
11
12 | X=dataset.drop(["Tarih","HKI"], axis=1)
13 | y=dataset["HKI"]
14 | # Feature Scaling (Özellik Ölçekleme)
15 | from sklearn.preprocessing import StandardScaler
16 | sc = StandardScaler()
17 | X = sc.fit_transform(X)
18 | # Veri setinin eğitim ve test olarak ayrılması
19 | from sklearn.model_selection import train_test_split
20 | X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)
21
22
23 |
24 | # Eğitim veri setine KNN sınıflandırıcısının uygulanması
25 | from sklearn.neighbors import KNeighborsClassifier
26 | classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
27 | classifier.fit(X_train, y_train)
28
29 | # Test verileri ile tahmin yapılması
30 | y_pred = classifier.predict(X_test)
31
32
33 | # Confusion matrisinin oluşturulması
34 | from sklearn.metrics import confusion_matrix
35 | cm = confusion_matrix(y_test, y_pred)
36
37 | #Accuracy (sınıflandırma için başarımlı metriği) ölçülmesi
38 | from sklearn.metrics import accuracy_score
39 | acc = accuracy_score(y_test, y_pred)
40 | print("Accuracy", acc)
41
42
43 | #f1_score (sınıflandırma için başarımlı metriği) ölçülmesi
44 | from sklearn.metrics import f1_score
45 | f1_score = f1_score(y_test, y_pred)
46 | print("f1 score", f1_score)
47
```

Görsel 3: KNeighborsClassifier (KNN)

Çıkan sonuç Görsel 4’de yer almaktadır.

```
In [1]: runfile('C:/Users/ASUS/Desktop/VeriBilimi_eski/Bayes.py',
wdir='C:/Users/ASUS/Desktop/VeriBilimi_eski')

In [2]: runfile('C:/Users/ASUS/Desktop/VeriBilimi_eski/KNN.py',
wdir='C:/Users/ASUS/Desktop/VeriBilimi_eski')
Accuracy 0.9239130434782609
f1 score 0.9542483660130718

In [3]:
```

Görsel 4: KNeighborsClassifier (KNN) Sonucu

GaussianNB (Naive Bayes)

Naive Bayes sınıflandırıcısının temeli Bayes teoremine dayanır. lazy (tembel) bir öğrenme algoritmasıdır aynı zamanda dengesiz veri kümelerinde de çalışabilir. Algoritmanın çalışma şekli bir eleman için her durumun olasılığını hesaplar ve olasılık değeri en yüksek olana göre sınıflandırır. Az bir eğitim verisiyle çok başarılı işler çıkartabilir. Test kümesindeki bir değer eğitim kümesinde gözlemlenemeyen bir değeri varsa olasılık değeri olarak 0 verir yani tahmin yapamaz. Bu durum genellikle Zero Frequency (Sıfır Frekans) adıyla bilinir. Bu durumu çözmek için düzeltme teknikleri kullanılabilir. En basit düzeltme tekniklerinden biri Laplace tahmini olarak bilinir. Kullanım alanlarına örnek olarak gerçek zamanlı tahmin, çok sınıflı tahmin, metin sınıflandırması, spam filtreleme, duyarlılık analizi ve öneri sistemleri verilebilir

Python’daki kodları Görsel 5’de yer almaktadır.

```
1 # Naive Bayes
2
3 # Kütüphanelerin import edilmesi
4 import numpy as np
5 import matplotlib.pyplot as plt
6 import pandas as pd
7
8 # Veri setinin okunması ve bağımsız-bağımlı değişkenlerin ayrılması
9 dataset = pd.read_csv('hki.csv')
10
11
12 X=dataset.drop(["Tarih","HKI"], axis=1)
13 y=dataset["HKI"]
14 # Feature Scaling (Özellik Ölçekleme)
15 from sklearn.preprocessing import StandardScaler
16 sc = StandardScaler()
17 X = sc.fit_transform(X)
18 # Veri setinin eğitim ve test olarak ayrılması
19 from sklearn.model_selection import train_test_split
20 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)
21
22 # Eğitim veri setine Naive Bayes sınıflandırıcısının uygulanması
23 from sklearn.naive_bayes import GaussianNB
24 classifier = GaussianNB()
25 classifier.fit(X_train, y_train)
26
27 # Test verileri ile tahmin yapılması
28 y_pred = classifier.predict(X_test)
29
30 # Confusion matrisinin oluşturulması
31 from sklearn.metrics import confusion_matrix
32 cm = confusion_matrix(y_test, y_pred)
33
34 #Accuracy (sınıflandırma için başarı metriği) ölçülmesi
35 from sklearn.metrics import accuracy_score
36 acc = accuracy_score(y_test, y_pred)
37 print("Accuracy", acc)
38
39 #f1_score (sınıflandırma için başarı metriği) ölçülmesi
40 from sklearn.metrics import f1_score
41 f1_score = f1_score(y_test, y_pred)
42 print("f1 score", f1_score)
```

Görsel 5: Naive Bayes

Çıkan sonuç Görsel 6’da yer almaktadır.

```
In [3]: runfile('C:/Users/ASUS/Desktop/VeriBilimi_eski/Bayes.py',
wdir='C:/Users/ASUS/Desktop/VeriBilimi_eski')
Accuracy 0.9130434782608695
f1 score 0.9459459459459458
```

Görsel 6: Naive Bayes Sonucu

RandomForestClassifier

Karar ağaçlarının en büyük problemlerinden biri aşırı öğrenme-veriyi ezberlemedir (overfitting). Random Forest modeli bu problemi çözmek için hem veri setinden hem de öznelik setinden Rastgele olarak 10'larca 100'lerce farklı alt-setler seçiyor ve bunları eğitiyor. Bu yöntemle 100'lerce karar ağacı oluşturuluyor ve her bir karar ağacı bireysel olarak tahminde bulunuyor.

Python’daki kodları Görsel 7’de yer almaktadır.

```
1 #- coding: utf-8 -*-
2 """
3 Created on Fri May 15 18:11:23 2020
4
5 @author: Gizem Çoban
6
7 """
8
9 #Random Forest
10
11 # Kütüphanelerin import edilmesi
12 import numpy as np
13 import matplotlib.pyplot as plt
14 import pandas as pd
15
16 # Veri setinin okunması ve bağımsız-bağımlı değişkenlerin ayrılması
17 dataset = pd.read_csv('hki.csv')
18
19
20 X=dataset.drop(["Tarih", "HKI"], axis=1)
21 y=dataset["HKI"]
22 # Feature Scaling (Özellik Ölçekleme)
23 from sklearn.preprocessing import StandardScaler
24 sc = StandardScaler()
25 X = sc.fit_transform(X)
26 # Veri setinin eğitim ve test olarak ayrılması
27 from sklearn.model_selection import train_test_split
28 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)
29
30 # Eğitim veri setine Random Forest sınıflandırıcısının uygulanması
31 from sklearn.ensemble import RandomForestClassifier
32 classifier = RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 0)
33 classifier.fit(X_train, y_train)
34
35
36 # Test verileri ile tahmin yapılması
37 y_pred = classifier.predict(X_test)
38
39
40 # Confusion matrisinin oluşturulması
41 from sklearn.metrics import confusion_matrix
42 cm = confusion_matrix(y_test, y_pred)
43
44 #Accuracy (sınıflandırma için başarıım metriği) ölçülmesi
45 from sklearn.metrics import accuracy_score
46 acc = accuracy_score(y_test, y_pred)
47 print("Accuracy", acc)
48
49
50 #f1_score (sınıflandırma için başarıım metriği) ölçülmesi
51 from sklearn.metrics import f1_score
52 f1_score = f1_score(y_test, y_pred)
53 print("f1 score", f1_score)
54
```

Görsel 7: RandomForestClassifier

Çıkan sonuç Görsel 8’de yer almaktadır.

```
In [8]: runfile('C:/Users/ASUS/Desktop/VeriBilimi_eski/Random
Forest.py', wdir='C:/Users/ASUS/Desktop/VeriBilimi_eski')
Accuracy 0.9891304347826086
f1 score 0.9934640522875817
```

Görsel 8: RandomForestClassifier Sonucu

SVC

Support Vector Machine sınıflandırma için kullanılan yöntemlerden birisidir. Temel olarak iki sınıflı bir doğru veya düzlem ile birbirinden ayırmaya çalışır. Bu ayırmayı da sınırdaki elemanlara göre yapar. SVC sınıflı scikit-learn kütüphanesi SVM modülünün altından projeye dahil edilmektedir.

Python’daki kodları Görsel 9’da yer almaktadır.

```
#!/usr/bin/env python
# coding: utf-8
"""
Created on Fri May 15 18:11:23 2020

@author: Gizem Çoban
"""

#SVC

# Kütüphanelerin import edilmesi
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Veri setinin okunması ve bağımsız-bağımlı değişkenlerin ayrılması
dataset = pd.read_csv('hki.csv')

X=dataset.drop(["Tarih","HKI"], axis=1)
y=dataset["HKI"]
# Feature Scaling (Özellik Ölçekleme)
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X = sc.fit_transform(X)
# Veri setinin eğitim ve test olarak ayrılması
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

# Eğitim veri setine SVM sınıflandırıcısının uygulanması
from sklearn.svm import SVC
classifier = SVC(kernel = 'rbf', random_state = 0)
classifier.fit(X_train, y_train)

# Test verileri ile tahmin yapılması
y_pred = classifier.predict(X_test)

# Confusion matrisinin oluşturulması
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

#Accuracy (sınıflandırma için başarımlı metriği) ölçülmesi
from sklearn.metrics import accuracy_score
acc = accuracy_score(y_test, y_pred)
print("Accuracy", acc)

#f1_score (sınıflandırma için başarımlı metriği) ölçülmesi
from sklearn.metrics import f1_score
f1_score = f1_score(y_test, y_pred)
print("f1 score", f1_score)
```

Görsel 9: SVC

Çıkan sonuç Görsel 10'da yer almaktadır.

```
In [9]: runfile('C:/Users/ASUS/Desktop/VeriBilimi_eski/SVC.py',
wdir='C:/Users/ASUS/Desktop/VeriBilimi_eski')
Accuracy 0.9456521739130435
f1 score 0.9668874172185431
```

Görsel 10: SVC Sonucu

Yapay Sinir Ağı (YSA)

Diğer bir tanıma göre yapay sinir ağları; insan beyninden esinlenerek geliştirilmiş, ağırlıklı bağlantılar aracılığıyla birbirine bağlanan ve her biri kendi belleğine sahip işlem elemanlarından oluşan paralel ve dağıtılmış bilgi işleme yapıları; bir başka deyişle, biyolojik sinir ağlarını taklit eden bilgisayar programlarıdır.

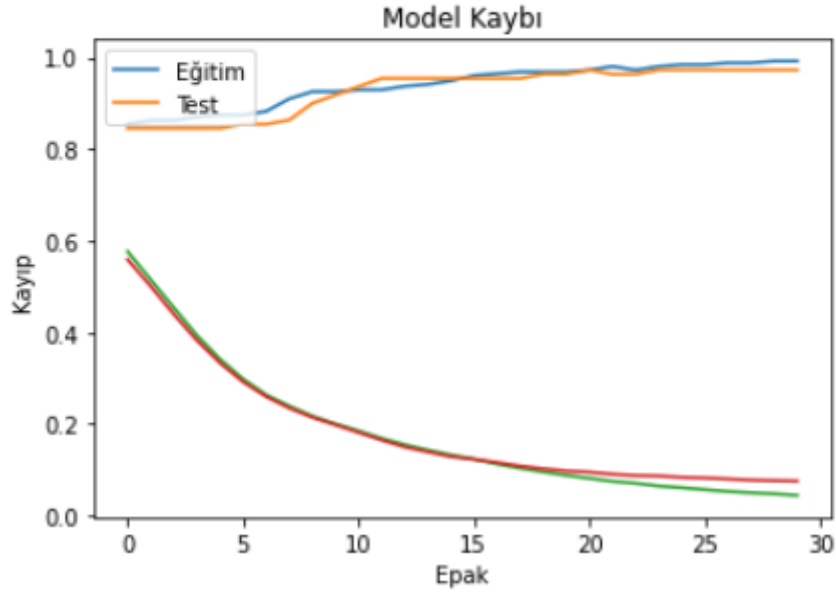
Python'daki kodları Görsel 11'de yer almaktadır.

```
7
8 import numpy as np
9 import pandas as pd
10 from sklearn.preprocessing import LabelEncoder
11
12 #Verisetinin yüklenmesi
13 veri=pd.read_csv("hki.csv")
14 #sınıf sayısını belirle
15 label_encoder=LabelEncoder().fit(veri.HKI)
16 labels=label_encoder.transform(veri.HKI)
17 classes=list(label_encoder.classes_)
18
19 x=veri.drop(["Tarih","HKI"], axis=1)
20 y=labels
21
22 #Verilerin standartlaştırılması
23 from sklearn.preprocessing import StandardScaler
24 sc=StandardScaler()
25 x=sc.fit_transform(x)
26
27 #Eğitim ve test verilerinin hazırlanması
28 from sklearn.model_selection import train_test_split
29 x_train, x_test, y_train, y_test=train_test_split(x,y,test_size=0.3) #%20sini testte kullanma
30
31 #Çıktı değerlerinin kategorileştirilmesi
32 from tensorflow.keras.utils import to_categorical
33 y_train=to_categorical(y_train)
34 y_test=to_categorical(y_test)
35
36 #YSA Modelinin oluşturulması
37 from tensorflow.keras.models import Sequential
38
39 #Dense ysa oluşturacağımız kısımdır.
40 from tensorflow.keras.layers import Dense
41
42 #16 tane nöron olsun. 20 tane alanımız bulunmakta. aktivasyonumuzda relu olsun
43 model=Sequential()
44 #girdi katmanı
45 model.add(Dense(16,input_dim=11,activation="relu"))
46 #ara katmanlar 3 tane eklendi
47 model.add(Dense(12,activation="relu"))
48 model.add(Dense(14,activation="relu"))
49 model.add(Dense(10,activation="relu"))
50 #çıkıti katmanı
51 #çıkıti sayısı kaçsa o kadar nöron olmak zorunda.
52 #aktivasyonu softmax olmak zorunda çünkü sınıflandırma yapılmaktadır.
53 model.add(Dense(2,activation="softmax"))
54 model.summary()
55
56 #Modelin Derlenmesi
57 model.compile(loss="binary_crossentropy",optimizer="adam",metrics=["accuracy"])
58
59 #modelin eğitilmesi epochs süresi40 verildi
60 model.fit(x_train,y_train,validation_data=(x_test,y_test),epochs=30)
61
```

Görsel 11: Yapay Sinir Ağı Modeli

Çıkan sonuç Görsel 12’da yer almaktadır.

```
Ortalama Eğitim Kaybı: 0.18049750563999017  
Ortalama Eğitim Başarımı: 0.9398692846298218  
Ortalama Doğrulama Kaybı: 0.18550443674127262  
Ortalama Doğrulama Başarımı: 0.9290909151236216  
<function matplotlib.pyplot.show>
```



Görsel 12: Yapay Sinir Ağı Modelinin Sonucu

