İSTANBUL
AYVANSARAY ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ

VERİ MADENCİLİĞİ DERSİ

''KİŞİSEL SAĞLIK VERİLERİ İÇEREN VERİ SETİNDE MALİYET TAHMİNİ''

Doç. Dr. Metin ZONTUL

Hazırlayan:

Samet ÖÇSOY
18040101008

13.01.2022

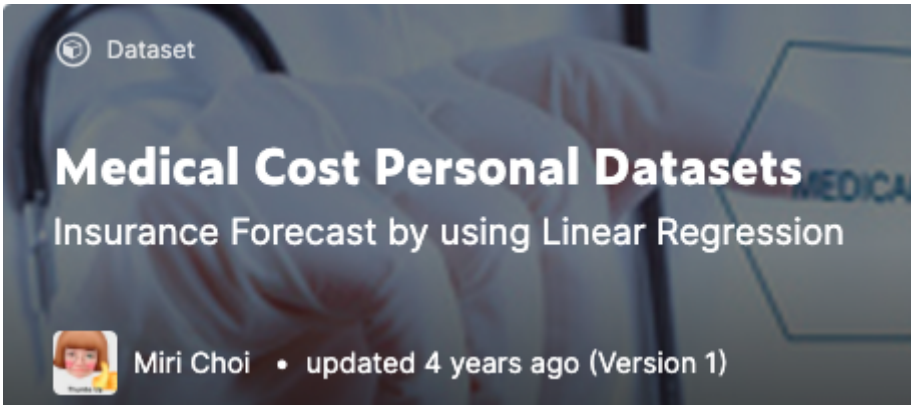
İçindekiler

1)Giriş

Doküman içeriğinde, makine öğrenmesinin çalışma alanlarından regresyon analizi konusuna ilişkin bir uygulama bulunmaktadır.

Bu projedeki amaç; içerisinde sağlık verileri bulunan bir veri seti üzerinde çalışma yaparak, bu çalışma sonucunda bir model oluşturmak ve oluşturulan modeli kullanarak **kişilerin sağlık maliyeti fiyatlarına yönelik bir tahmin değeri** üretmektir.

Projede kullanılan veri seti:



insurance.csv [1]

2)Veri Seti İçerisindeki Öznitelikler (Features) :

age: Yaş değeri

sex: Medeni durum değeri (evli-bekar)

bmi: Vücut kitle endeksi değeri

children: Çocuk sayısı değeri

smoker: Sigara içip içmediğine ilişkin değer

region: Bölge değeri

charges: Tahmin edilecek olan masraf değeri

3)Uygulama Kodları

3.1)Kütüphaneler

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LinearRegression, Ridge, Lasso,
ElasticNet

from sklearn.metrics import mean_squared_error, r2_score,
mean_absolute_error
```

Açıklama:

Numpy: Matris işlemleri için kullanılmaktadır.

Pandas: Veri ön işleme aşaması ve veri manipülasyonu için kullanılmaktadır.

Matplotlib: Görselleştirme için kullanılmaktadır.

Scikit-learn: Makine öğrenmesi aşaması için kullanılmaktadır.

3.2) Veri Setini Okuma

```
# Veri Setinini Okuyoruz
data_set = pd.read_csv("insurance.csv")
print(data_set.to_string())
```

 # Çıktı:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.924000
1	18	male	33.770	1	no	southeast	1725.552300
2	28	male	33.000	3	no	southeast	4449.462000
3	33	male	22.705	0	no	northwest	21984.470610
4	32	male	28.880	0	no	northwest	3866.855200
5	31	female	25.740	0	no	southeast	3756.621600
6	46	female	33.440	1	no	southeast	8240.589600
7	37	female	27.740	3	no	northwest	7281.505600
8	37	male	29.830	2	no	northeast	6406.410700
9	60	female	25.840	0	no	northwest	28923.136920
10	25	male	26.220	0	no	northeast	2721.320800
11	62	female	26.290	0	yes	southeast	27808.725100
12	23	male	34.400	0	no	southwest	1826.843000
13	56	female	39.820	0	no	southeast	11090.717800
14	27	male	42.130	0	yes	southeast	39611.757700
15	19	male	24.600	1	no	southwest	1837.237000
16	52	female	30.780	1	no	northeast	10797.336200
17	23	male	23.845	0	no	northeast	2395.171550
18	56	male	40.300	0	no	southwest	10602.385000
19	30	male	35.300	0	yes	southwest	36837.467000
20	60	female	36.005	0	no	northeast	13228.846950
21	30	female	32.400	1	no	southwest	4149.736000
22	18	male	34.100	0	no	southeast	1137.011000
23	34	female	31.920	1	yes	northeast	37701.876800
24	37	male	28.025	2	no	northwest	6203.901750
25	59	female	27.720	3	no	southeast	14001.133800

Veri setine ilk olarak bir göz atıldığında verilerin etiketli bir veri olduğunu gözlemleyebiliyoruz.

Makine öğrenmesi;

1 - Etiketli veriler için **Gözetimli Öğrenme,**
2 - Etiketsiz veriler için **Gözetimsiz Öğrenme**

olmak üzere ikiye ayrılmaktadır.

Not: Bunun dışında **Pekiştirmeli Öğrenme** tipi de mevcuttur.

Gözetimli öğrenme, kendi içerisinde iki başlığa ayrılmaktadır:

a)**Regresyon Analizi**

Sayısal veriler üzerinde analiz ve makine öğrenmesi modeli aşamaları gerçekleştirilerek aynı şekilde **bir sayısal değer tahmin etme sürecidir.**

Örnek: (Dolar Kuru Tahmini, Ev Fiyatları Tahmini, Hisse Senedi Tahmini vb.)

b)**Sınıflandırma Analizi**

Sayısal veriler üzerinde analiz ve makine öğrenmesi modeli aşamaları gerçekleştirilerek **bir kategorik değer tahmin etme sürecidir.**

Tabi ki her ne kadar kategorik verileri tahmin etmeye çalışsakta bu değerleri de sayısal forma dönüştürmeli ve o form da çalışmalıyız.

Çünkü makine öğrenmesi sadece sayısal veriler üzerinde çalışır. Veri setinde kategorik veri olabilir ancak bu veriler sayısal veriye dönüştürülür.

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.924000
1	18	male	33.770	1	no	southeast	1725.552300
2	28	male	33.000	3	no	southeast	4449.462000
3	33	male	22.705	0	no	northwest	21984.470610
4	32	male	28.880	0	no	northwest	3866.855200
5	31	female	25.740	0	no	southeast	3756.621600
6	46	female	33.440	1	no	southeast	8240.589600
7	37	female	27.740	3	no	northwest	7281.505600
8	37	male	29.830	2	no	northeast	6406.410700
9	60	female	25.840	0	no	northwest	28923.136920
10	25	male	26.220	0	no	northeast	2721.320800
11	62	female	26.290	0	yes	southeast	27808.725100
12	23	male	34.400	0	no	southwest	1826.843000
13	56	female	39.820	0	no	southeast	11090.717800
14	27	male	42.130	0	yes	southeast	39611.757700
15	19	male	24.600	1	no	southwest	1837.237000
16	52	female	30.780	1	no	northeast	10797.336200
17	23	male	23.845	0	no	northeast	2395.171550
18	56	male	40.300	0	no	southwest	10602.385000
19	30	male	35.300	0	yes	southwest	36837.467000
20	60	female	36.005	0	no	northeast	13228.846950
21	30	female	32.400	1	no	southwest	4149.736000
22	18	male	34.100	0	no	southeast	1137.011000
23	34	female	31.920	1	yes	northeast	37701.876800
24	37	male	28.025	2	no	northwest	6203.901750
25	59	female	27.720	3	no	southeast	14001.133800

Veri setinin incelediğimizde **‘‘charges’’** değişkeninin etiket verisi olduğu ve sayısal bir veri formatında olduğu görülmektedir.

Buradan diğer değişkenler üzerinden çalışmalar yapılarak **‘‘charges’’** değişkeninin tahmin edilmeye çalışılacağını kısacası bu problemin bir **Regresyon Analizi** olduğu yorumunu yapabiliriz.

3.3)Veri Ön İşleme Aşaması

a) Bilgi Almak

```
print(data_set.describe())
```

```
 age          bmi     children       charges
count  1338.000000  1338.000000  1338.000000   1338.000000
mean     39.207025    30.663397     1.094918  13270.422265
std      14.049960     6.098187     1.205493  12110.011237
min      18.000000    15.960000     0.000000   1121.873900
25%      27.000000    26.296250     0.000000   4740.287150
50%      39.000000    30.400000     1.000000   9382.033000
75%      51.000000    34.693750     2.000000  16639.912515
max      64.000000    53.130000     5.000000  63770.428010
```

İlk olarak veri setinde bir keşifçi analiz yapıyoruz. **describe()** fonksiyonu ile veri setine ilişkin bilgileri görüntülüyoruz.

Sonuçlara baktığımızda veri setimizdeki tüm kolonlardan sadece birkaçının sadece burada gösterildiğini görüyoruz. Buradan kategorik verileri sayısal formata dönüştürmemiz gerektiğini anlıyoruz.

---

b) Eksik Veri Analizi

```
print(data_set.isnull().sum())
```

Eksik verileri görmek için **isnull().sum()** komutunu kullanıyoruz.

 # Çıktı

```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

Sonuçlar incelendiğinde veri setinde hiçbir verinin eksik olmadığı görülmektedir.

---

c) Verilerin Dağılım Durumunun Analizi

```
age_statistic = data_set["age"].value_counts()
sex_statistic = data_set["sex"].value_counts()
bmi_statistic = data_set["bmi"].value_counts()
children_statistic = data_set["children"].value_counts()
smoker_statistic = data_set["smoker"].value_counts()
region_statistic = data_set["region"].value_counts()

print("yaş kolonundaki değerkerin kaç defa tekrarladıkları :
\n",age_statistic)
print("\n\n")
print("medeni durum degerlerinin kac defa tekrarladiklari :
\n",sex_statistic)
print("\n\n")
print("vucut kitle indeksi durumlarinin kac defa tekrarladiklari :
\n",bmi_statistic)
print("cocuk sayisi verilerinin kac defa tekrarladiklari :
\n",children_statistic)
print("\n\n")
print("sigara icip icmediklerine iliskin degerlerinin kac defa
tekrarladiklari : \n",smoker_statistic)
print("bolge ozniteligine iliskin verilerin kac defa tekrarladiklari :
|n",region_statistic)
```

 # Çıktılar

```
Medeni durum değerlerinin kac defa tekrarladıkları :
 male      676
female    662
Name: sex, dtype: int64
```

```
Vücut kitle indeksi durumlarının kac defa tekrarladıkları :

 32.300    13
28.310     9
31.350     8
30.800     8
30.875     8
          ..
20.100     1
29.545     1
37.900     1
25.520     1
24.090     1
Name: bmi, Length: 548, dtype: int64
```

```
Çocuk sayısı verilerinin kaç defa tekrarladıkları :

 0    574
1    324
2    240
3    157
4     25
5     18
Name: children, dtype: int64
```

```
Sigara içip içmediklerine ilişkin değerlerin kaç defa tekrarladıkları :

 no     1064
yes     274
Name: smoker, dtype: int64
```

Bölge özniteliğine ilişkin verilerin kaç defa tekrarladıkları :

```
southeast    364
southwest    325
northwest    325
northeast    324
Name: region, dtype: int64
```

Verilere bakıldığında genel olarak verilerin dengeli dağıldığı görülmektedir. Burada verilerin sadece belirli bir kısımda ağırlıklı olarak toplandığı bir durum oluşması durumunda ise o kolonda bir düzenlemeye ihtiyaç duyulacağı söylenebilir.

---

d) Aykırı Veri Analizi

```
sex_types = pd.get_dummies(data_set.sex, prefix='sex')
smoker_types = pd.get_dummies(data_set.smoker, prefix='smoker')
region_types = pd.get_dummies(data_set.region, prefix='region')

data_set = pd.concat([data_set, sex_types, smoker_types, region_types],
axis=1)

print(data_set.to_string())

data_set.drop(['sex', 'smoker', 'region', 'sex_female', 'smoker_no'],
axis=1, inplace = True)

print(data_set.to_string())
```

Kategorik verileri pd.get_dummies() fonksiyonunu kullanarak **ONE-HOT-ENCODİNG** işlemi ile Nominal verilere dönüştürüyoruz.

Sonrasında ise önceden kolan kategorik veri kolonları ve dönüşüm sonrası oluşan kukla değişkenler veri setinden temizlemektedir.

 # Çıktı

age	bmi	children	charges	sex_male	smoker_yes	region_northeast	region_northwest	region_southeast	region_southwest
0	19	27.900	0	16884.924000	0	1	0	0	0	1
1	18	33.770	1	1725.552300	1	0	0	0	1	0
2	28	33.000	3	4449.462000	1	0	0	0	1	0
3	33	22.705	0	21984.470610	1	0	0	1	0	0
4	32	28.880	0	3866.855200	1	0	0	1	0	0
5	31	25.740	0	3756.621600	0	0	0	0	1	0
6	46	33.440	1	8240.589600	0	0	0	0	1	0
7	37	27.740	3	7281.505600	0	0	0	1	0	0
8	37	29.830	2	6406.410700	1	0	1	0	0	0
9	60	25.840	0	28923.136920	0	0	0	1	0	0
10	25	26.220	0	2721.320800	1	0	1	0	0	0
11	62	26.290	0	27808.725100	0	1	0	0	1	0
12	23	34.400	0	1826.843000	1	0	0	0	0	1
13	56	39.820	0	11090.717800	0	0	0	0	1	0
14	27	42.130	0	39611.757700	1	1	0	0	1	0
15	19	24.600	1	1837.237000	1	0	0	0	0	1

Sonuç olarak veri ön işleme aşamasında veri setimizi makine öğrenmesi modeline hazır hale getirmiş oluyoruz.

3.4)Verilerin Ayrıştırılması

a) Bağımlı ve Bağımsız Ayrıştırması

```
y = data_set["charges"]

data_set.drop(["charges"], axis=1, inplace=True)

x = data_set
```

Veri setini x (bağımlı) ve y (bağımsız) olmak üzere iki bölüme ayırıyoruz.

b) Eğitim ve Test Ayrıştırması

```
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.25, random_state=46)
```

Verileri;

x_train
x_test
y_train
y_test

olmak üzere 4 parçaya ayırıyoruz.

Not: test_size değeri 0.25 olarak ayarlanmıştır. Bu değer daha sonra değiştirilerek model performansında iyileştirme yapılabilmektedir.

3.5)Normalizasyon Aşaması

```
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.fit_transform(x_test)
```

Sonuç;

Bu kısımda veriler arasındaki değer farklarından dolayı verilerin birbirlerini yutma ihtimaline karşın veriler standardize ederek küçük bir değer aralığında sıkıştırıyoruz.

3.6)Makine Öğrenmesi Aşaması

```
# 4.1 - KARAR AGACI REGRESYON ALGORITMASI

tree_regression = DecisionTreeRegressor(random_state=42, max_depth=2)
tree_regression = tree_regression.fit(x_train, y_train)
tahmin_tree_regression = tree_regression.predict(x_test)


# 4.2 - RANDOM FORREST REGRESYON ALGORITMASI

random_regression = RandomForestRegressor(max_depth=4, random_state=42)
random_regression.fit(x_train, y_train)
tahmin_random_regression = random_regression.predict(x_test)


# 4.3 - LASSO REGRESYON ALGORITMASI

lassoReg = Lasso(alpha=2)
lassoReg.fit(x_train,y_train)
tahmin_lasso = lassoReg.predict(x_test)


# 4.4 - Elastic REGRESYON ALGORITMASI

elastic_reg = ElasticNet(random_state=0)
elastic_reg.fit(x_train, y_train)
tahmin_elastic = elastic_reg.predict(x_test)


# 4.5 RIDGE REGRESYON ALGORITMASI

ridge_reg = Ridge()
ridge_reg.fit(x_train, y_train)
tahmin_ridge = ridge_reg.predict(x_test)
```

SONUÇ

Bu kısımda sırasıyla;

1 - **Karar Ağacı Regresyon Algoritması,**

2 - **Rastgele Orman Regresyon Algoritması,**

3 - **Lasso Regresyon Algoritması,**

4 - **Elastik Regresyon Algoritması,**

5 - **Ridge Regresyon Algoritması**

üzerinde eğitim ve tahmin işlemleri yapılmıştır.

3.7)Performans Değerlendirme Aşaması

```
# SONUCLARI HESAPLAMA FONKSIYONU
def performance_calculate(predict):
   mae = mean_absolute_error(y_test, predict)
   mse = mean_squared_error(y_test, predict)
   rmse = np.sqrt(mse)  # or mse**(0.5)
   r2 = r2_score(y_test, predict)

   data = [mae, mse, rmse, r2]

   return data
```

Bu kısımda fonksiyon, modellerin tahmin değerlerini parametre alarak aşağıdaki performans değerlerini hesaplamaktadır.

a)**Mean Absolute Error** (Ortalama Mutlak Hata)

$$\text{Mean Absolute Error} = \frac{1}{N}\sum_{i=0}^{n} |y_i - \hat{y}_i|$$

b)**Mean Squared Error** (Ortalama Kare Hata)

$$MSE = \frac{1}{N}\sum_{(x,y)\in D} (y - prediction(x))^2$$

c)**Root Mean Squared Error** (Kök Ortalama Kare Hata)

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

d)**R Squared** (R Kare)

$$R^2 = 1 - \frac{Explained\ Variation}{Total\ Variation}$$

Metriklerin yorumlanması;

R-Kare değerinin büyük olması modelin doğruluğu açısından avantajdır. Bu nedenle karşılaştırma yaparken bu değerin büyük olma durumunu inceleyeceğiz.

Diğer metriklerin ise R-Kare'nin aksine küçük olması beklenir. Ne kadar küçük bir değer bulunursa model burada daha iyi sonuç vereceği anlamına gelmektedir. Veri setlerine bağlı olarak metriklerin önemi ve yorumlanması değişmektedir.

3.7.1) Performans Sonuçlarını Ekrana Yazdırma

```
predicts =
[tahmin_tree_regression,tahmin_random_regression,tahmin_lasso,
tahmin_elastic, tahmin_ridge]

algoritma_names = ["Karar Ağacı Regresyon", "Random Forrest regresyon",
"Lasso Regresyon", "Elastic Regresyon", "Ridge Regresyon"]

# EKRANA YAZDIRMAK
seriler = []
metrics = ["Mean Absolute Error (MAE)", "Mean Squared Error (MSE)",
"Root Mean Squared Error (RMSE)" , "R2"]

for i in predicts:
   data = performance_calculate(i)
   seriler.append(data)

from IPython.display import HTML

df = pd.DataFrame(data = seriler, index = algoritma_names, columns =
metrics)
pd.set_option('display.colheader_justify', 'center') # kolon isimlerini
ortaliyoruz

print(df.to_string())
```

 # Çıktı

	(MAE)	(MSE)	(RMSE)	(R2)
Karar Ağacı Regresyon	3299.054220	2.383989e+07	4882.611455	0.824404
Rassal Orman Regresyon	2622.613819	1.812672e+07	4257.549152	0.866485
Lasso Regresyon	4201.461867	3.645130e+07	6037.491554	0.731513
Elastic Regresyon	5000.126534	4.595553e+07	6779.051012	0.661508
Ridge Regresyon	4202.696230	3.644396e+07	6036.883183	0.731567

4)Değerlendirme

Rassal Orman Regresyonun bu veri seti için diğer algoritmalara karşı daha iyi olduğunu söyleyebiliriz. Ancak rassal orman regresyonun değerinin de hala geliştirilmesi gerekmektedir.

Lasso ve Ridge regresyon için aynı sonuçları verdiği yorumunu yapabiliriz.

Karar ağacı algoritması ise normal şartlarda regresyon algoritmalarında çok da iyi bir algoritma olmamasına rağmen bu veri seti için nispeten iyi bir sonuç vermiştir.

Elastic regresyon ise algoritmalar arasından en verimsiz çalışanı olmuştur.

Model performanslarında iyileştirme yapmak için, korelasyon analizi ve daha başarılı bir veri ön işleme aşaması gerçekleştirilmelidir.

5) Kaynakça

[1]
https://www.kaggle.com/mirichoi0218/insurance/code

[2]
Uygulama Kodları:
https://github.com/asmtttt/regression_analysis_medical_dataset_in_python