Yapay Sinir Ağları

ÖMER FARUK TAŞCAN 170101047

MAHMUT FURKAN YILDIRIM 170101040

-YAPAY SİNİR AĞLARI İLE MÜŞTERİNİN BİR SONRAKİ AY İÇİN TEMERRÜT ÖDEMESİ OLUP OLMAYACAĞINI SINIFLANDIRMA-

2022

İçindekiler

1.GİRİŞ

Bu projenin amacı, verileri görselleştirmek ve ön gözlemler sağlamaktır. Bundan sonra, regresyon için NeuralNet sınıfının miras alınması ve Doğrusal Olmayan Regresyon(Non-Linear Regression) sınıflandırıcısının uygulanması ve yapı dahil olmak üzere iyi sinir ağı parametrelerini bulmak ve test doğruluğunu rapor etmek için 5 katlı çapraz doğrulama kullanılması gerekir. Bunu başarmak için Lojistik Regresyon sınıfı ve 5 katlı çapraz doğrulama işlevi oluşturuldu ve ardından modeli eğitip test doğruluğunu raporlandı.

2.DATA

2.1 Dataset Bilgisi : Bu araştırma, Tayvan Chung Hua Üniversitesi'nde yapılmıştır ve müşterilerin kredi kartı temerrüt ödemelerini amaçlamaktadır. Amaç, müşterinin bir sonraki ay için temerrüt ödemesi olup olmayacağını sınıflandırmaktır. Veri kümesi, kararın bağlı olacağı 23 girdiye sahiptir.

Aşağıdaki attributes(nitelik)'lerden oluşur :

1.default payment(Yes=1, No=0)

2.X1: Amount of the given credit(Verilen kredi tutarı)

3.X2: Gender(1=Male, 2= Female)

4.X3: Education(1 = graduate school; 2 = university; 3 = high school; 4 = others).

5.X4: Marital Status(1=married, 2= single ,3=others)

6.X5 : Age(year)

7.X6-X11 : History of past payment. Tracked the past monthly payment records (from April to September, for the year 2005) as follows:
X6 = the repayment status in September, 2005; X7 = the repayment status in August, 2005; . . .;X11 = the repayment status in April, 2005.
The measurement scale for the repayment status is: -1 = pay duly, -2 = No consumption; 1 = payment delay for one month; 2 = payment delay for two months; . . .; 8 = payment delay for eight months; 9 = payment delay for nine months and above.

8. X12-X17 : Amount of bill statement .

X12 = amount of bill statement in September, 2005; X13 = amount of bill statement in August, 2005; . . .; X17 = amount of bill statement in April, 2005.

9. X18-X23 : Amount of previous payment.
X18 = amount paid in September, 2005; X19 = amount paid in August, 2005; . . .;X23 = amount paid in April, 2005.

2.2 Source of Data : **https://archive.ics.uci.edu/ml/machine-learning-databases/00350/**

-Gerekli kütüphanelerin import edilmesi

```
import math
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib
import copy
import matplotlib.pyplot as plt

from sklearn import metrics
from IPython.display import display, Markdown, HTML

# Configurations
matplotlib.rcParams['agg.path.chunksize'] = 10000
%matplotlib inline

# Set display parameters
pd.set_option('display.float_format', lambda x: '%.3f' % x)
pd.set_option('display.max_rows', 15)
pd.set_option('display.max_columns', 10)
```

2.3 Verisetinin okunması

```
df = pd.read_csv(r'Credit Card Defaulters.csv')
df.head()
```

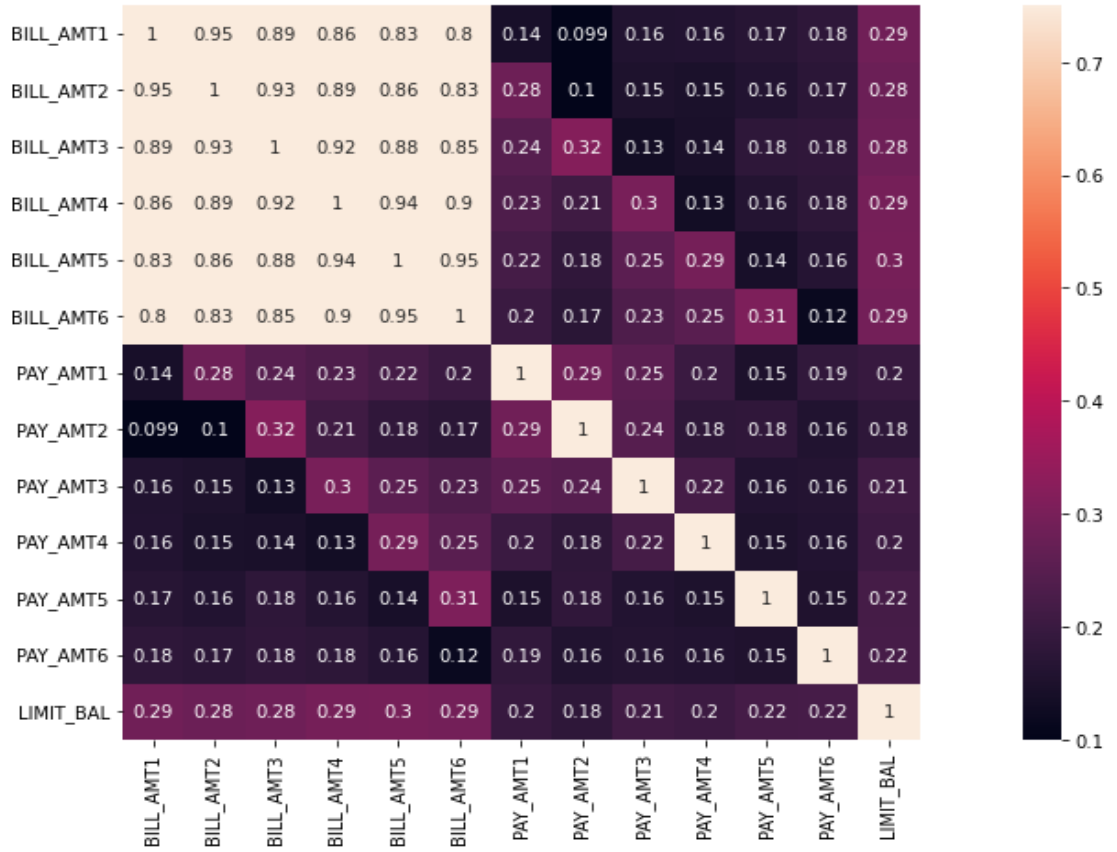
2.4 Verisetinin daha anlaşılır olması için düzenleme yapılması

```
#Grafiklerin görsel olarak daha çekici ve okunabilir görünmesi
#için sayıyı karakterlerle değiştiren ön işlemenin bir parçası

df['SEX'].replace({1:'Male', 2:'Female'},inplace=True)
df['MARRIAGE'].replace({1:'Married',2:'Single',3:'Others'},inplace=True)
df['EDUCATION'].replace({1:'Gradudate School',2:'Univeristy',3:'High School',4:'
df.head()
```

	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	...	PAY_AMT3	PAY_AMT4	PAY_AMT5	PAY_A
0	1	20000	Female	Univeristy	Married	...	0	0	0	
1	2	120000	Female	Univeristy	Single	...	1000	1000	0	2
2	3	90000	Female	Univeristy	Single	...	1000	1000	1000	5
3	4	50000	Female	Univeristy	Married	...	1200	1100	1069	
4	5	50000	Male	Univeristy	Married	...	10000	9000	689	

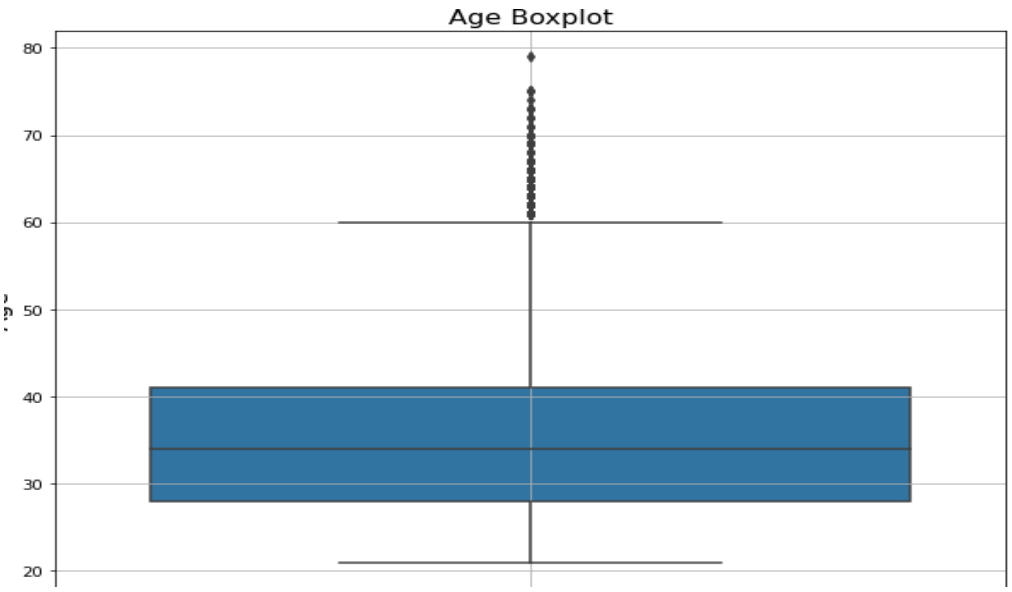
5 rows x 25 columns

2.5 Korelasyon Matrisi

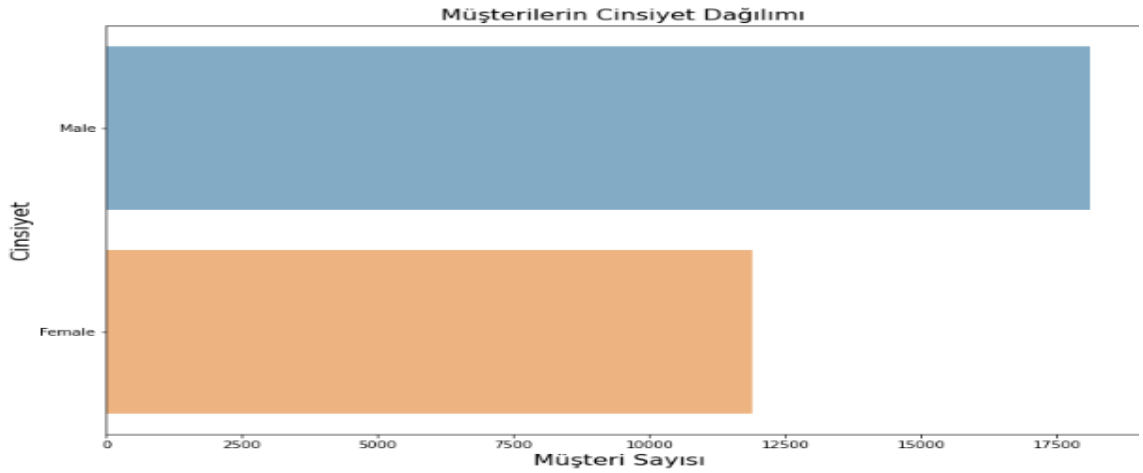


Limit bakiyeleri, fatura tutarları ve ödeme tutarları arasında bir korelasyon matrisi oluşturuldu. Limit bakiyeleri ile ödeme ve fatura tutarları arasında düşük bir korelasyon olduğu söylenebilir. Ancak senetlerin katlanmış tutarları yansıtması nedeniyle beklendiği gibi senet tutarlarının birbirleri arasında yüksek bir korelasyona sahip olduğu görülmektedir.

2.6 BoxPlot : Medyan yaş 34 ve çoğu müşteri 28-41 yaş aralığındadır.

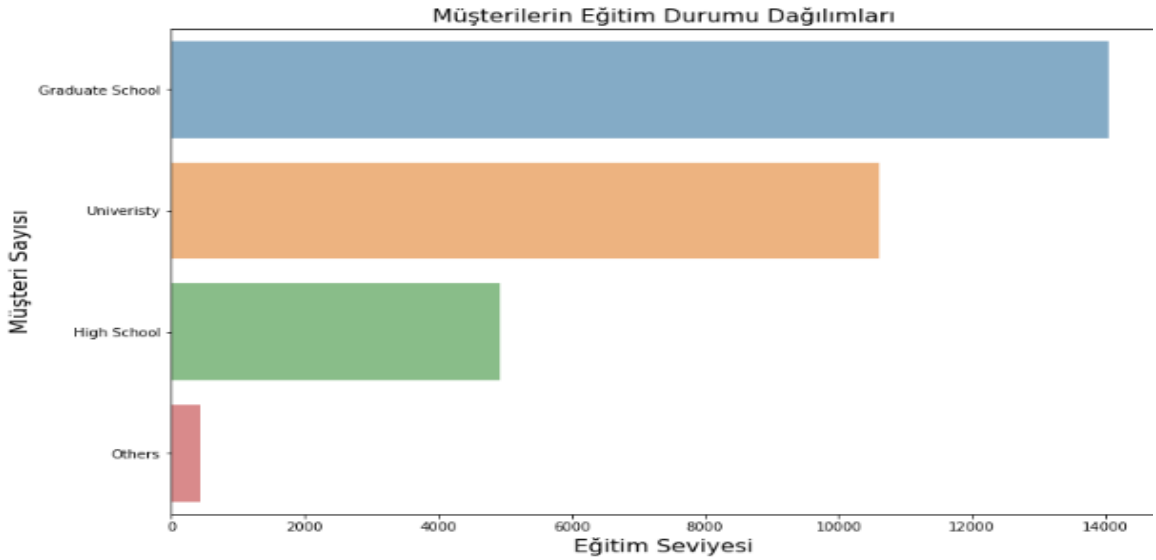


2.6.1 BarPlot



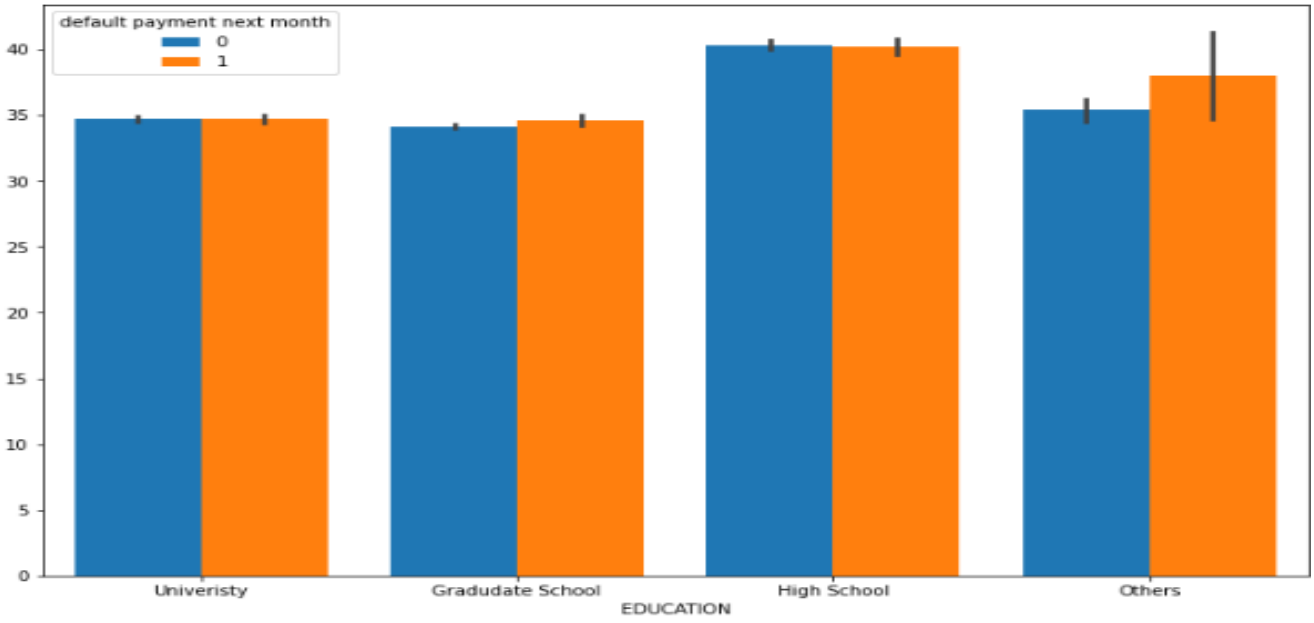
Yukarıdaki barplot, veri kümesindeki erkek ve kadın müşterilerin sayısını göstermektedir. Veri setinde erkek müşteriler çoğunlukta olduğundan, erkek müşterilerin kadın müşterilere göre kredi kartını daha fazla kullanma ihtimalinin olabileceği sonucuna varılabilir.

2.6.2 Barplot :



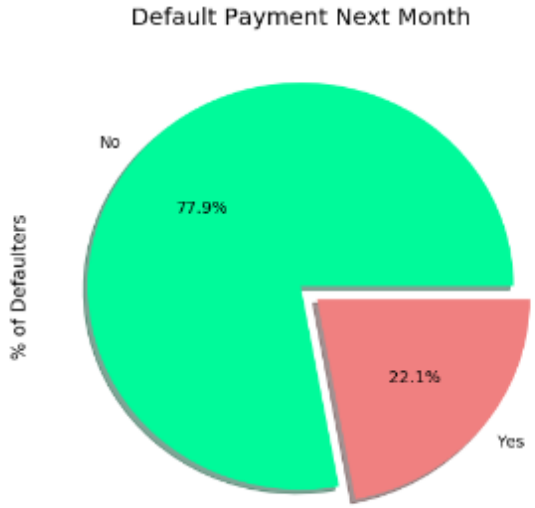
Yukarıdaki barplot, veri setindeki müşteri sayısını ve eğitim seviyelerini göstermektedir. Kredi kartını en fazla Enstitü öğrencisi kullanmakta, bunu üniversiteye giden öğrenciler, lise öğrencileri ve diğerleri izlemektedir.

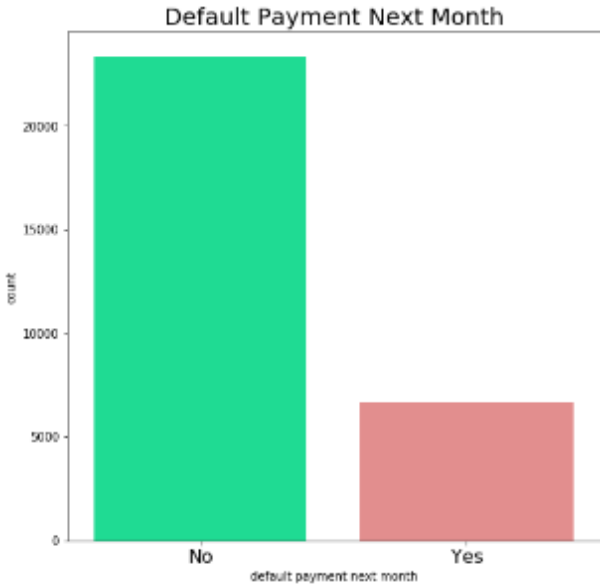
2.6.4 Barplot



Yukarıdaki grafik, yaş ve eğitim arasındaki bir karşılaştırmadır ve gelecek ay varsayılan ödemesi olan müşterileri gösterir. Temerrüde düşenlerin çoğu, muhtemelen okuldan ayrılan ve bu nedenle ödemelerini karşılamak için burs şeklinde fonları olmayan ve sonunda temerrüde düşenler olabilecek diğer kişilerdir.

2.7 Pie Chart and Count Plot :



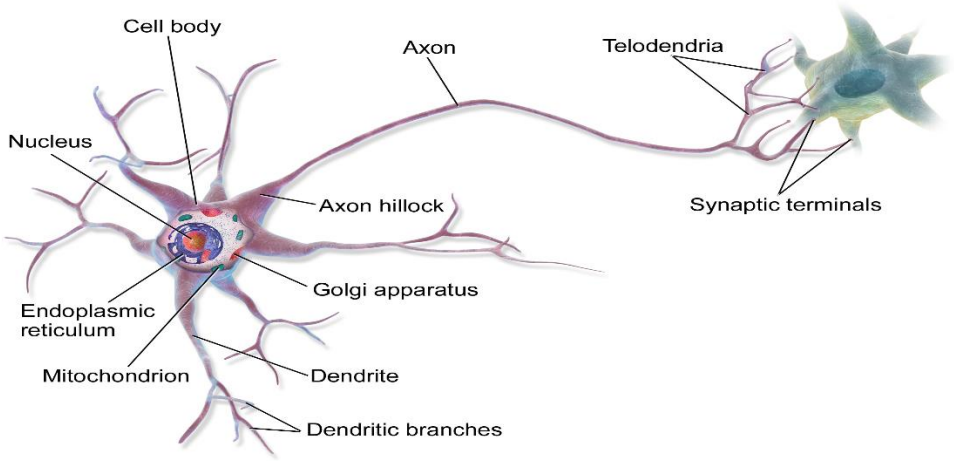


Pasta grafiği temerrüde düşenlerin(borcun zamanında ödenmemesi veya belirtilen tutarın tamamını ödemediği takdirde oluşan durum) yüzdesini verir.Toplam 30.000 müşteriden 6.630 u yani %22,1 lik kısmının gelecek ay temerrüde düşeceği görüülmektedir.Geriye kalan %77,9 yani 23.370 kişi ödemesini zamanında yapacağı anlaşılmaktadır.

3.Metotlar

3.1 Neural Network : Neural Network , bilgisayarlara verileri insan beyninden esinlenerek işlemeyi öğreten bir yapay zeka yöntemidir. Bu, insan beynine benzeyen katmanlı bir yapıda birbirine bağlı düğümleri veya nöronları kullanan ve derin öğrenme adı verilen bir tür makine öğrenimi sürecidir. Bilgisayarların hatalarından ders çıkarmak ve sürekli olarak gelişmek için kullandığı uyarlanabilir bir sistem oluşturur. Böylece, yapay sinir ağları, belgelerin özetini çıkarma ya da yüzleri tanıma gibi karmaşık sorunları daha fazla doğrulukla çözmeye çalışır.

Aşağıda bir nöronun anatomisi gösterilmiştir.



Yapay Sinir Ağlarının Kullanım Alanları

- **Hesaplamalı Finans(Computational finance):** Kredi skorlaması(credit scoring), Algoritmik Ticaret(algorithmic trading)
- **Görüntü işleme ve bilgisayarla görü(image processing and computer vision):** Yüz tanıma(face recognition), hareket tanıma(motion detection), nesne tanıma(object detection)
- **Hesaplamalı biyoloji(Computational biology):** Tümör bulma(tumor detection), İlaç keşfi(durg discovery), DNA dizilimi(DNA seqencing)
- **Enerji üretimi (Energy production):** Fiyat ve yük tahmini (price and load forecasting)
- **Otomotiv, havacılık ve üretim (Automotive, aerospace and manufacturing):** Öngörücü bakım (predictive maintenance)
- **Doğal dil işleme(Natural language processing):** Sesli Asistan(voice assistant), Duygu analizi(emotion analysis)

3.2 Nonlinear Logistic Regression (Doğrusal Olmayan Regresyon)

Doğrusal olmayan regresyon, regresyon modelinin bir bağımlı değişken ile bağımsız değişkenler arasında doğrusal olmayan bir ilişkiyi tasvir ettiği bir regresyon analizi anlamına gelir. Başka bir deyişle, yordayıcı ve yanıt değişkeni arasındaki ilişki doğrusal olmayan bir model izler.

Doğrusal olmayan model karmaşıktır ve aynı zamanda doğru sonuçlar verir.Analiz sağlanan veri kümesine dayalı olarak değişkenler arasındaki ilişkiyi gösteren bir eğri geliştirir. Büyük esneklik sunan model, senaryoya en uygun eğriyi oluşturabilir. Nonlinear Regression matematiksel olarak ifade etmek istenirse ;

$$g_k(x) = P(T = k \mid x) = \frac{e^{\kappa_k}}{\sum_{c=1}^{K} e^{\kappa_c}}$$

Bu softmax işlevini kullanarak, tüm sınıflar için olasılıksal çıktılar üretildi. Çok etiketli sınıfları işlemek için, doğrusal model için ağırlıkları güncellemek üzere eğitim için gösterge hedef etiketlerini kullanılır. Türetmeyi takiben, aşağıdaki güncelleme kuralına ulaşıldı :

$$w_j \leftarrow w_j + \alpha \sum_{n=1}^{N} \Big(t_{n,j} - g_j(x_n) \Big) x_n.$$

Batch örnekleri ile ağırlıkları güncellemek için, bu güncelleme kuralını matris formuna aşağıdaki gibi dönüştürebiliriz:

$$w \leftarrow w + \alpha X^{\top} \Big(T - g(X) \Big).$$

Daha önce türetme için aşağıdaki hata fonksiyonundan başlanması gerekir :

$$E(w) = -\ln P(T \mid w) = -\sum_{n=1}^{N} \sum_{k=1}^{K} t_{n,k} \ln y_{n,k}.$$

Şimdi, bunu iki katmanlı sinir ağlarına genişletilir. Sinir ağının regresyon için türetilmesine benzer şekilde, gradyanı yukarıdaki negatif log olabilirlik fonksiyonu ile karelenmiş hatayı değiştirerek türetilebilir.

$E(\mathbf{w})$ hata fonksiyonundan, her katman için ağırlıkları güncellemek için gradyanı türetilebilir .

$$v_{dg} \leftarrow v_{dg} - \alpha_h \frac{\partial E(\mathbf{W}, \mathbf{V})}{\partial v_{dg}}$$

$$w_{gk} \leftarrow w_{gk} - \alpha_o \frac{\partial E(\mathbf{W}, \mathbf{V})}{\partial w_{gk}},$$

burada $\boldsymbol{\alpha h}$ ve $\boldsymbol{\alpha o}$ sırasıyla gizli ve çıkış katmanı için öğrenme oranıdır. Burada sinir ağının çıktısını κ olarak gösterilir .

$$
\begin{aligned}
\frac{\partial E}{\partial w_{gk'}} &= -\frac{\partial\left(\frac{1}{N}\frac{1}{K}\sum_{n=1}^{N}\sum_{k=1}^{K}(t_{nk}\ln g_{nk}(\mathbf{x}_n))\right)}{\partial w_{gk'}} \\
&= -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{n,k}\frac{1}{g_{n,k'}(\mathbf{x}_n)}\frac{\partial g_{n,k'}(\mathbf{x}_n)}{\partial w_{gk'}} \\
&= -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{n,k}\frac{1}{g_{n,k'}(\mathbf{x}_n)}\frac{\partial g_{n,k'}(\mathbf{x}_n)}{\partial \kappa_{nk}}\frac{\partial \kappa_{nk}}{\partial w_{gk'}} \\
&= -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{n,k}\frac{1}{g_{n,k'}(\mathbf{x}_n)} g_{nk'}(\mathbf{x}_n)(I_{kk'} - g_{nk'}(\mathbf{x}_n))\frac{\partial \kappa_{nk}}{\partial w_{gk'}} \\
&= -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{n,k}(I_{kk'} - g_{nk'}(\mathbf{x}_n))\frac{\partial \sum_{g=0}^{G} z1_{ng} w_{gk}}{\partial w_{gk'}} \\
&= -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{n,k}(I_{kk'} - g_{nk'}(\mathbf{x}_n)) z1_{nk'} \\
&= -\sum_{n=1}^{N}\left(\sum_{k=1}^{K} t_{n,k}(I_{kk'} - g_{nk'}(\mathbf{x}_n))\right) z1_{nk'} \\
&= -\sum_{n=1}^{N}\left(\sum_{k=1}^{K} t_{n,k} I_{kk'} - g_{nk'}(\mathbf{x}_n)\sum_{k=1}^{K} t_{n,k}\right) z1_{nk'} \\
&= -\sum_{n=1}^{N}\left(t_{n,k'} - g_{nk'}(\mathbf{x}_n)\right) z1_{nk'} .
\end{aligned}
$$

Bu gradyanı matris formuna dönüştürmek ve ağırlık güncellememize yansıtmak

$$\mathbf{W} \leftarrow \mathbf{W} + \alpha_o \mathbf{Z1}^{\top}\Big(\mathbf{T} - g(\mathbf{X})\Big).$$

Şimdi gizli katman için ağırlığı v güncellenir. Gizli katman için şu işlem tekrarlanır:

$$
\begin{aligned}
\frac{\partial E}{\partial v_{dg}} &= \frac{\partial\left(\frac{1}{N}\frac{1}{K}\sum_{n=1}^{N}\sum_{k=1}^{K}(t_{nk}\ln g_{nk}(\mathbf{x}_n))\right)}{\partial v_{dg}} \\
&= -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{n,k}\frac{1}{g_{n,k}(\mathbf{x}_n)}\frac{\partial g_{n,k}(\mathbf{x}_n)}{\partial v_{dg}} \\
&= -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{n,k}\frac{1}{g_{n,k}(\mathbf{x}_n)}\frac{\partial g_{n,k}(\mathbf{x}_n)}{\partial \kappa_{nk}}\frac{\partial \kappa_{nk}}{\partial v_{dg}} \\
&= -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{n,k}\frac{1}{g_{n,k}(\mathbf{x}_n)}\frac{\partial g_{n,k}(\mathbf{x}_n)}{\partial \kappa_{nk}}\sum_{g=0}^{G} w_{gk}\frac{\partial h(a_{ng})}{\partial a_{ng}}\frac{\partial a_{ng}}{\partial v_{dg}} \\
&= -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{n,k}(I_{kk'} - g_{nk'}(x_n))\sum_{g=0}^{G} w_{gl}\frac{\partial h(a_{ng})}{\partial a_{ng}}\frac{\partial a_{ng}}{\partial v_{dg}} \\
&= -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{n,k}(I_{kk'} - g_{nk'}(\mathbf{x}_n))\sum_{g=0}^{G} w_{gk}\frac{\partial h(a_{ng})}{\partial a_{ng}} x1_{nd} \\
&= -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{n,k}(I_{kk'} - g_{nk'}(\mathbf{x}_n))\sum_{g=0}^{G} w_{gk}\frac{\partial h(a_{ng})}{\partial a_{ng}} x1_{nd} \\
&= -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{n,k}(I_{kk'} - g_{nk'}(\mathbf{x}_n))\sum_{g=0}^{G} w_{gk}(1 - z_{ng}^{2}) x1_{nd} .
\end{aligned}
$$

Yine gizli ağırlık güncellemesi için matris formunda dönüştürme,

$$\mathbf{V} \leftarrow \mathbf{V} + \alpha_h \mathbf{X1}^{\top}\Big((\mathbf{T} - g(\mathbf{X}))\mathbf{W}^{\top} \odot (1 - \mathbf{Z}^{2})\Big).$$

3.3 Preliminary Analysis NonLinear Logistic Regression

(Doğrusal Olmayan Lojistik Regresyonda Ön Analiz)

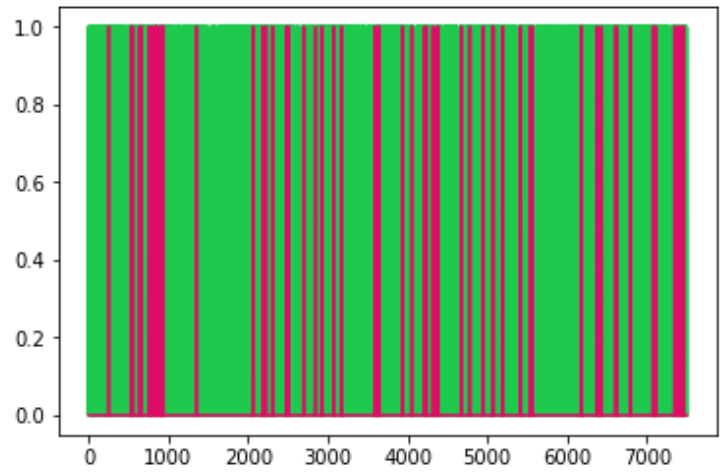
Önce çapraz doğrulama olmadan hedef test setinin doğruluğunu hesaplanır ve daha sonra 5 kat çapraz doğrulamanın hedef test setinin doğruluğunu artırıp artırmayacağını görülür. Bunun için 4 tane gizli alındı ve epochlar 500 kez eğitildikten sonra bu sınıflandırıcıyı train setinde kullanılır ve hedef test setinin doğruluğu kontrol edilir.

```
clsf = NeuralNetLogReg([X_train.shape[1], 4, 2]) #Classifier takes input as Non
T_train = T_train.reshape(T_train.shape[0],1)
clsf.train(X_train, T_train, ftracep=True)
y = clsf.use(X_test)
```

```
y= np.argmax(y, 1)
```

```
plt.plot(T_test, color = "#1ECA4E")
plt.plot(y, color = "#DE0D67")

print("Accuracy: ", 100 - np.mean(np.abs(T_test - y)) * 100, "%")
```

```
Accuracy:  77.27074844444445 %
```



3.4 Logistic Regression

Lojistik regresyon, bir veri setinin önceki gözlemlerine dayanarak evet veya hayır gibi ikili bir sonucu tahmin etmek için istatistiksel bir analiz yöntemidir.

Bir lojistik regresyon modeli , bir veya daha fazla mevcut bağımsız değişken arasındaki ilişkiyi analiz ederek bağımlı bir veri değişkenini tahmin eder. Örneğin, bir siyasi adayın bir seçimi kazanıp kazanamayacağını veya bir lise öğrencisinin belirli bir koleje kabul edilip edilmeyeceğini tahmin etmek için bir lojistik regresyon kullanılabilir. Bu ikili sonuçlar, iki alternatif arasında basit kararlara izin verir.

-Seçilen Sınıflandırıcı Liner Regresyon. Lojistik regresyon modeli her k sınıfı için $P(T=k|\mathbf{x})$ tahmin eder. Doğrusal Regresyon modelini yeniden ele almak:

$$\kappa=f(\mathbf{x};\mathbf{w})=\mathbf{X}\mathbf{w}.$$

-Böylece Lojistik Regresyon için elde :

$$P(T=k|\mathbf{x})=h(\mathbf{X}\mathbf{w})=h(\kappa)=y.$$

-Lojistik Regresyon için hata fonksiyonu şu şekilde tanımlanır:

$$E(\mathbf{w})=-\ln P(\mathbf{T}|\mathbf{w})=-\sum n=1N\sum k=1Ktn,k\ln yn,k.$$

Aynı zamanda cross-entropy hata fonksiyonu olarak da adlandırılır.

-Bu hata fonksiyonunda Gradient Descent uygulamak:

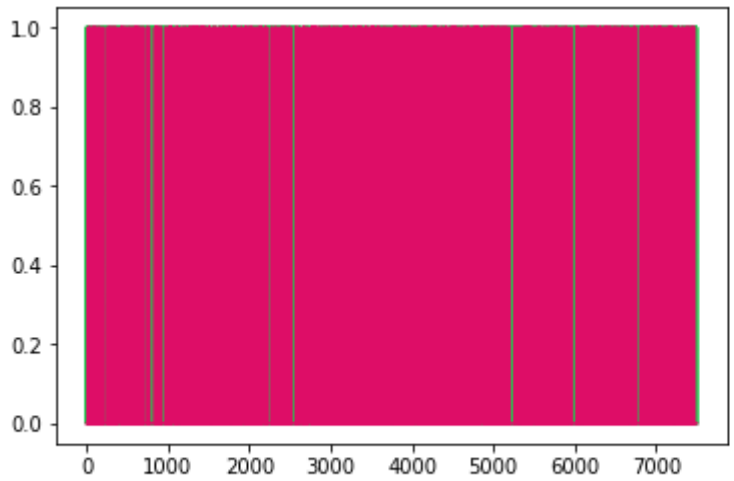
$$\mathbf{w}_j \leftarrow \mathbf{w}_j + \alpha\mathbf{X}^\top\left(t_{*j} - g_j(\mathbf{X})\right).$$

3.4.1 Preliminary Analysis Logistic Regression

Önce hedef test setinin doğruluğunu çapraz doğrulama olmadan hesaplanır ve daha sonra 5 kat çapraz doğrulamanın hedef test setinin doğruluğunu artırıp artırmayacağını test edilir. Bunun için alfa öğrenme oranımı 0.1 olarak alınır ve epoch'lar 500 kez eğitildi, ardından bu sınıflandırıcıyı train setinde kullanılır ve hedef test setinin doğruluğu kontrol edilir .

```
alpha_learning_rate= [0.1 , 500]
clsf = LogisticRegression(alpha_learning_rate)
T_train = T_train.reshape(-1,1)
clsf.train(X_train,T_train)
y = clsf.use(X_test)
y_max = np.argmax(y, 1)
```

```
plt.plot(T_test, color = "#1ECA4E")
plt.plot(y_max, color = "#DE0D67")
plt.show()

print("Accuracy: ", 100 - np.mean(np.abs(T_test - y_max)) * 100, "%")
```




```
Accuracy:  73.67996444444444 %
```

3.5 5-Fold Cross Validation

5 kat çapraz doğrulama işlevi oluşturuldu. Çapraz doğrulama, orijinal numuneyi modeli eğitmek için bir eğitim setine ve onu değerlendirmek için bir test setine bölerek öngörücü modelleri değerlendirmek için kullanılan bir tekniktir. 5 katlı çapraz doğrulamada, orijinal numune rastgele 5 eşit boyutlu alt numuneye bölünür. 5 alt örnekten tek bir alt örnek, modeli test etmek için doğrulama verileri olarak tutulur ve geri kalan 4 alt örnek, eğitim verileri olarak kullanılır. Çapraz doğrulama işlemi daha sonra 5 kez (kıvrımlar) tekrarlanır ve 5 alt numunenin her biri tam olarak bir kez doğrulama verisi olarak kullanılır. Bu yöntemin avantajı, tüm gözlemlerin hem eğitim hem de doğrulama için kullanılması ve her gözlemin doğrulama için tam olarak bir kez kullanılmasıdır.

Eğitim setinde bağımsız değişkenler olan X_train ve bağımlı değişken olan T_train olarak giriş parametrelerini alan 5 katlı çapraz doğrulama için bir işlev tanımlandı.

Test bölümleri ve doğrulama bölümleri oluşturulurve ardından tüm train maskelenir. Bundan sonra train sınıflandırıcı oluşturuldu ve ardından test setini maskelendi.

4. Sonuçlar

4.1 5-Fold Cross Validation ile Logistic Regression

Öncelikle Lojistik Regresyon sınıflandırıcısını 5-kat çapraz doğrulama ile eğitilecek ve ardından Lojistik regresyon için en iyi yapı parametresini bulunacaktır. Bunun için en iyi argümanı alacak olan test_err adında bir fonksiyon oluşturuldu .

4.1.1 Deney 1

Parameter Choice(Parametre Seçimi) : Deney 1 için, parametreler alındı, alfa 0,1 ve 0,01 ve veri trainlerinin sayısı 500. Bu modelde sağlanan parametreler arasından en iyi parametreyi bulacak ve grafikleri çizecek ve doğruluğu hesaplayacak, f- skor, konfüzyon matrisi ve ROC eğrisi.

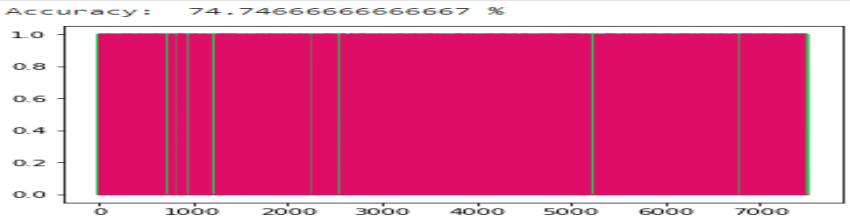
Accuracy :

```
# find best parameter
minErr = float('inf')
for test_err in test_errs:
    if test_err[1] < minErr:
        best_arg = test_err[0]
        minErr = test_err[1]

model = LogisticRegression(best_arg)
model.train(X_train, T_train)
Y = model.use(X_test)
```

```
c = np.argmax(Y, axis=1).reshape(-1, 1)
```

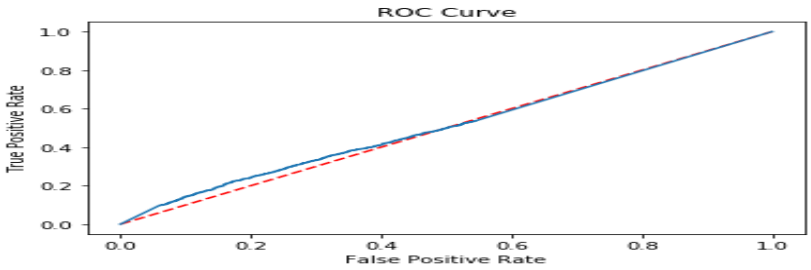
```
plt.plot(T_test, color = "#1ECA4E")
plt.plot(c, color = "#DE0D67")
print("Accuracy: ", 100 - np.mean(np.abs(T_test - c)) * 100, "%")
```

```
Accuracy:  74.746666666666667 %
```

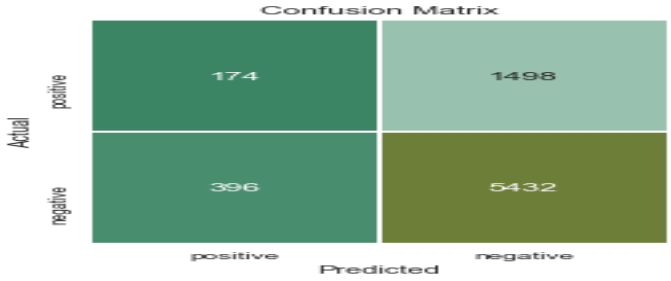


ROC Curve

```
Text(0,0.5,'True Positive Rate')
```



Confusion Matrix



4.1.2 Deney 2

Deney 2 için, parametreler alındı, alfa aynı olacak, ancak eğitimi ve veri katarlarının sayısını 1000 artırıldı. Bunda model, sağlanan parametreler arasından en iyi parametreyi bulacak ve grafikleri çizecek ve doğruluğu hesaplayacaktır. , f-skoru, karışıklık matrisi ve ROC eğrisi. Ayrıca her iki deneyin sonuçlarını da karşılaştırılır.

Accuracy

```
alpha_learning_rate = [(0.1, 1000), (0.01, 1000) ]
test_errs = cross_val_5fold(X_train, T_train, alpha_learning_rate, LogisticRegres
```

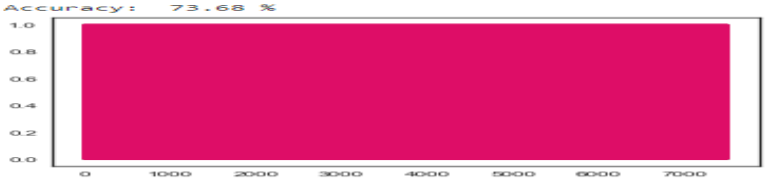
```
22500
[    0  4500  9000 13500 18000]
```

```
# find best parameter
minErr = float('inf')
for test_err in test_errs:
    if test_err[1] < minErr:
        best_arg = test_err[0]
        minErr = test_err[1]

model = LogisticRegression(best_arg)
model.train(X_train, T_train)
Y = model.use(X_test)
```

```
c = np.argmax(Y, axis=1).reshape(-1, 1)
```

```
plt.plot(T_test, color = "#1ECA4E")
plt.plot(c, color = "#DE0D67")
print("Accuracy: ", 100 - np.mean(np.abs(T_test - c)) * 100, "%")
```

```
Accuracy:  73.68 %
```



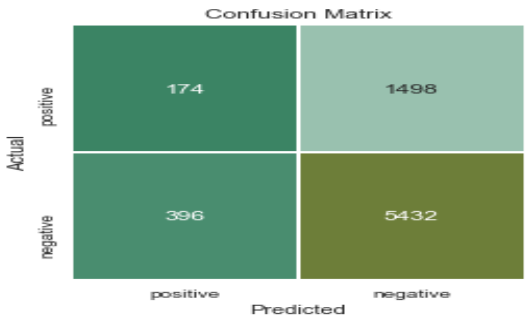
Accuracy,AUC, f-score, precision ,recall ,support

```
P = Y[:, 1].reshape(-1, 1)

# ROC Curve
fpr, tpr, thresh = metrics.roc_curve(T_test, P, pos_label=1)


# Prepare statistics Dataframe
stats = dict()
stats['accuracy'] = metrics.accuracy_score(T_test, c)
stats['precision'], stats['recall'], stats['f-score'], stats['support'] = \
metrics.precision_recall_fscore_support(T_test, c)
stats['auc'] = np.trapz(tpr, fpr)
stats = pd.DataFrame(stats, index=['negative', 'positive']).round(2)
stats
```

	accuracy	auc	f-score	precision	recall	support
negative	0.750	0.510	0.850	0.780	0.930	5828
positive	0.750	0.510	0.160	0.310	0.100	1672

ROC Curve

Confusion Matrix



Sınıflandırma Sonuçları : Alfayı aynı tutup model üzerindeki eğitim arttırıldığı zaman doğruluğunda arttığı gözlemlenmiştir.

4.2 Neural Network with 5-fold Cross Validation

Öncelikle Lojistik Regresyon sınıflandırıcısını 5 kat çapraz doğrulama ile eğitilecek ve ardından Lojistik regresyon için en iyi yapı parametresi bulunacaktır. Bunun için test_err adında en iyi argümanı alacak bir fonksiyon oluşturuldu .

4.2.1 Deney 1

Network Structure Choice(Ağ yapısı seçimi)

Bunun için model iki gizli katmana alınır ve verilerin accuracy, f-score , confusion matrix ve ROC eğrisinin ne olduğu bulunur .

```
nunits = [[23, 3, 2], [23, 4, 2]]
test_er = cross_val_5fold(X_train, T_train, nunits, NeuralNetLogReg)
```

```
22500
[    0  4500  9000 13500 18000]
```

```
# find best parameter
minErr = float('inf')
for test_err in test_er:
    if test_err[1] < minErr:
        best_arg = test_err[0]
        minErr = test_err[1]

model = NeuralNetLogReg(best_arg)
model.train(X_train, T_train)
Y = model.use(X_test)
```

```
c = np.argmax(Y, axis=1).reshape(-1, 1)
```

```
plt.plot(T_test, color = "#1ECA4E")
plt.plot(c, color = "#DE0D67")
print("Accuracy: ", 100 - np.mean(np.abs(T_test - c)) * 100, "%")
```

```
P = Y[:, 1].reshape(-1, 1)
# ROC Curve
fpr, tpr, thresh = metrics.roc_curve(T_test, P, pos_label=1)


# Prepare statistics Dataframe
stats = dict()
stats['accuracy'] = metrics.accuracy_score(T_test, c)
stats['precision'], stats['recall'], stats['f-score'], stats['support'] = \
metrics.precision_recall_fscore_support(T_test, c)
stats['auc'] = np.trapz(tpr, fpr)
stats = pd.DataFrame(stats, index=['negative', 'positive']).round(2)
stats
```

```
sns.set(style="white")
fig.add_subplot(1, 3, 1)
cmap = sns.diverging_palette(150, 100, as_cmap=True)
sns.heatmap(cm, fmt="d", cmap=cmap, annot=True, square=True, cbar=False, linewidths=.5)
plt.title('Confusion Matrix')
plt.ylabel('Actual');
plt.xlabel('Predicted');
```

```
# ROC Curve
fig.add_subplot(1, 3, 3)
plt.plot([0,1],[0,1],'r--')
plt.plot(fpr, tpr)
plt.title('ROC Curve')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
```

4.2.2 Deney 2

Network Structure Choice(Ağ yapısı seçimi)

Bunun için modelin doğruluğunu nasıl etkilediğini görmek için üçüncü bir gizli katman ekledim ve doğruluk, f-skoru, karışıklık matrisi ve ROC eğrisi sonuçlarını deney 1 ile karşılaştıracağım.

```
nunits = [[23, 3, 2], [23, 4, 2], [23, 5, 2]]
test_er = cross_val_5fold(X_train, T_train, nunits, NeuralNetLogReg)
```

```
# find best parameter
minErr = float('inf')
for test_err in test_er:
    if test_err[1] < minErr:
        best_arg = test_err[0]
        minErr = test_err[1]

model = NeuralNetLogReg(best_arg)
model.train(X_train, T_train)
Y = model.use(X_test)
```

```
c = np.argmax(Y, axis=1).reshape(-1, 1)
```

```
plt.plot(T_test, color = "#1ECA4E")
plt.plot(c, color = "#DE0D67")
print("Accuracy: ", 100 - np.mean(np.abs(T_test - c)) * 100, "%")
```

```
P = Y[:, 1].reshape(-1, 1)
# ROC Curve
fpr, tpr, thresh = metrics.roc_curve(T_test, P, pos_label=1)


# Prepare statistics Dataframe
stats = dict()
stats['accuracy'] = metrics.accuracy_score(T_test, c)
stats['precision'], stats['recall'], stats['f-score'], stats['support'] = \
metrics.precision_recall_fscore_support(T_test, c)
stats['auc'] = np.trapz(tpr, fpr)
stats = pd.DataFrame(stats, index=['negative', 'positive']).round(2)
stats
```

```
sns.set(style="white")
fig.add_subplot(1, 3, 1)
cmap = sns.diverging_palette(150, 100, as_cmap=True)
sns.heatmap(cm, fmt="d", cmap=cmap, annot=True, square=True, cbar=False, linewidths=.5)
plt.title('Confusion Matrix')
plt.ylabel('Actual');
plt.xlabel('Predicted');
```

```
# ROC Curve
fig.add_subplot(1, 3, 3)
plt.plot([0,1],[0,1],'r--')
plt.plot(fpr, tpr)
plt.title('ROC Curve')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
```

Sınıflandırma Sonuçları

Deney 1 ve deney 2'den de göründüğü gibi, ağ yapısına gizli bir katman eklendiğinde modelin doğruluğu artar, çünkü daha fazla katman ekledikçe boyutsallık artar ve problemin karmaşıklığı azalır .

Classfication model Comparison (Sınıflandırma modeli karşılaştırması)

Yukarıdaki sonuçlardan da göründüğü gibi Doğrusal Olmayan Lojistik Regresyon Modeli, Lojistik Regresyondan daha iyidir, bunun nedeni, her katmanın bir önceki katmana istediğiniz herhangi bir işlevi uygulayabildiği gizli katmanlardır, gizli katmanın işi girdileri bir şeye dönüştürmektir. çıktı katmanının kullanabileceği ve çıktı katmanının gizli katman aktivasyonlarını çıktınızın olmasını istediğiniz ölçeğe dönüştürdüğü. Dolayısıyla, Doğrusal Olmayan Lojistik Regresyon Modelinin doğruluğu, 5 kat çapraz doğrulama uygulandıktan sonra Lojistik Regresyondan daha iyidir.

Classfication Results Summary (Sınıflandırma Sonuçları Özeti)

Lojistik Regresyon ve Doğrusal Olmayan Lojistik Regresyon modellerine 5 kat çapraz doğrulama uygulandıktan sonra sınıflandırma sonuçları iyileşti. 5-katlı çapraz doğrulama tahmincisi, mevcut veri miktarı sınırlıysa çok önemli olabilen, tek bir uzatma kümesi tahmin edicisinden daha düşük bir varyansa sahiptir. Verilerin %90'ının eğitim için ve %10'unun test için kullanıldığı tek bir bekletme kümeniz varsa, test kümesi çok küçüktür, dolayısıyla farklı veri örnekleri için performans tahmininde çok fazla değişiklik olacaktır. veya eğitim ve test setleri oluşturmak için verilerin farklı bölümleri için. 5 katlı doğrulama, k farklı bölümün ortalamasını alarak bu varyansı azaltır, böylece performans tahmini, verilerin bölümlenmesine karşı daha az duyarlı olur. Tekrarlanan k-katlı çapraz doğrulama ile daha da ileri gidebiliriz; burada çapraz doğrulama, k alt küme oluşturmak için verilerin farklı bölümlenmesi kullanılarak gerçekleştirilir ve ardından bunun üzerinden de ortalama alınır.

5.KAYNAKÇA

- https://www.techtarget.com/searchbusinessanalytics/definition/logistic-regression#:~:text=Logistic%20regression%20is%20a%20statistical,or%20more%20existing%20independent%20variables.

- https://www.wallstreetmojo.com/nonlinear-regression/

- https://www.investopedia.com/terms/n/nonlinear-regression.asp

- https://www.veribilimiokulu.com/yapay-sinir-agiartificial-neural-network-nedir/#:~:text=Yapay%20sinir%20a%C4%9Flar%C4%B1%20(YSA)%2C,geli%C5%9Ftirilen%20bilgisayar%20sistemleridir%5B1%5D.

- https://aws.amazon.com/tr/what-is/neural-network/

- https://www.baeldung.com/cs/k-fold-cross-validation

- https://visualstudiomagazine.com/articles/2013/10/01/understanding-and-using-kfold.aspx