MAKİNE ÖĞRENMESİ

VERİSETLERİNDE

(ISIC_2019_Training_GroundTruth.csv)
(ISIC_2019_Training_Metadata.csv)
GRAFİK OLUŞTURMA,LOGİSTİC REGRESSİON,
NAİVE BAYES ,SUPPORT VECTOR MACHİNE(SVM)

KULLANARAK SINIFLANDIRMA

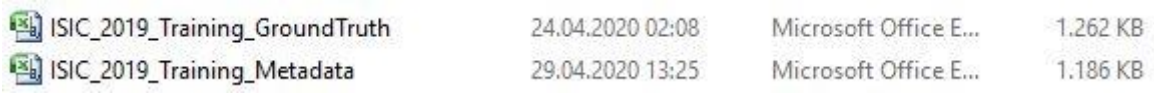
ALPER ŞAHİN ÖNER
BİLGİSAYAR MÜHENDİSİ

alpersahinoner@gmail.com

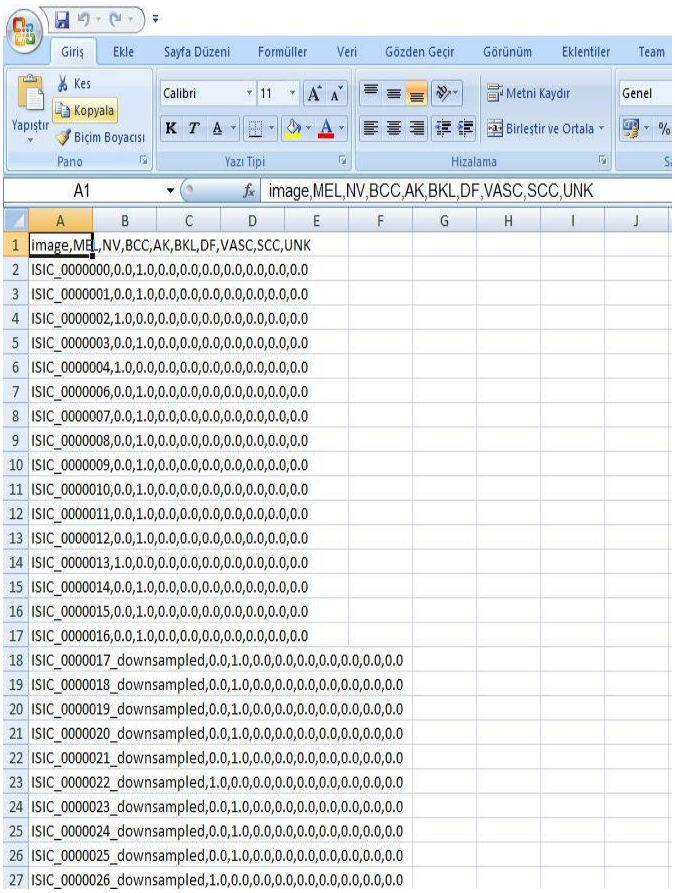
İçindekiler

1-Dataset incelemesi

Bu veri kümesi deri kanseri(melonama) ve bu hastalığa yakalanan,yakalanmayan kişiler hakkında bilgileri içermektedir.



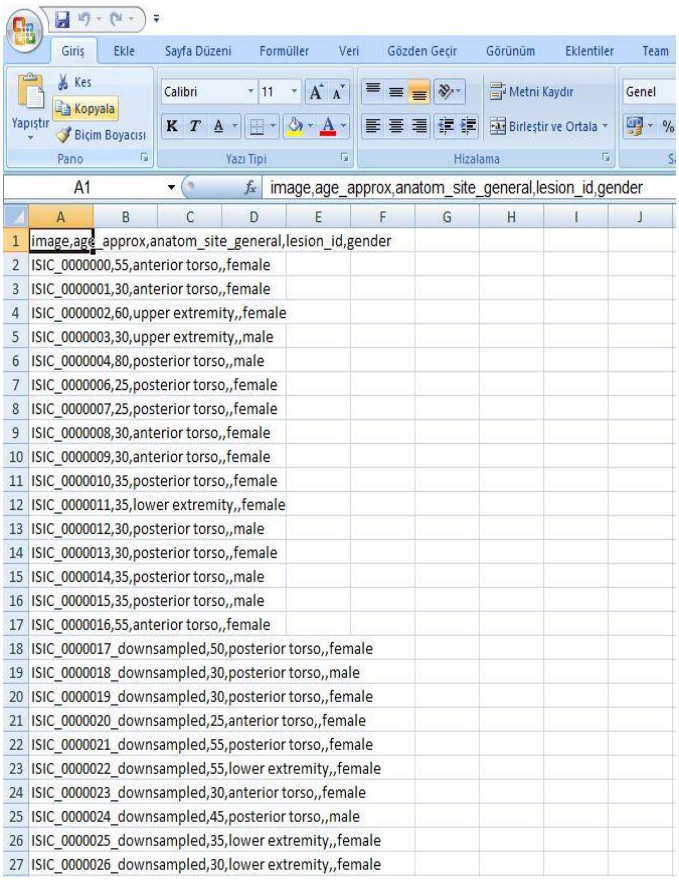
ISIC_2019_Training_GroundTruth.csv



	A
1	image,MEL,NV,BCC,AK,BKL,DF,VASC,SCC,UNK
2	ISIC_0000000,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
3	ISIC_0000001,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
4	ISIC_0000002,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
5	ISIC_0000003,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
6	ISIC_0000004,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
7	ISIC_0000006,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
8	ISIC_0000007,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
9	ISIC_0000008,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
10	ISIC_0000009,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
11	ISIC_0000010,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
12	ISIC_0000011,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
13	ISIC_0000012,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
14	ISIC_0000013,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
15	ISIC_0000014,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
16	ISIC_0000015,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
17	ISIC_0000016,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
18	ISIC_0000017_downsampled,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
19	ISIC_0000018_downsampled,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
20	ISIC_0000019_downsampled,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
21	ISIC_0000020_downsampled,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
22	ISIC_0000021_downsampled,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
23	ISIC_0000022_downsampled,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
24	ISIC_0000023_downsampled,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
25	ISIC_0000024_downsampled,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
26	ISIC_0000025_downsampled,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
27	ISIC_0000026_downsampled,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0

Fotoğrafların birbirinden ayırt etmek için 'image' sütününda adları belirtilmiş ve hasta olan kişiler 'MEL' sütününda 1 olarak gösterilmektedir.Diğer sütünlar hastalığın ayrıntılarını içermektedir.Toplamda 25332 tane satır vardır.

ISIC_2019_Training_Metadata.csv



	A
1	image,age_approx,anatom_site_general,lesion_id,gender
2	ISIC_0000000,55,anterior torso,,female
3	ISIC_0000001,30,anterior torso,,female
4	ISIC_0000002,60,upper extremity,,female
5	ISIC_0000003,30,upper extremity,,male
6	ISIC_0000004,80,posterior torso,,male
7	ISIC_0000006,25,posterior torso,,female
8	ISIC_0000007,25,posterior torso,,female
9	ISIC_0000008,30,anterior torso,,female
10	ISIC_0000009,30,anterior torso,,female
11	ISIC_0000010,35,posterior torso,,female
12	ISIC_0000011,35,lower extremity,,female
13	ISIC_0000012,30,posterior torso,,male
14	ISIC_0000013,30,posterior torso,,female
15	ISIC_0000014,35,posterior torso,,male
16	ISIC_0000015,35,posterior torso,,male
17	ISIC_0000016,55,anterior torso,,female
18	ISIC_0000017_downsampled,50,posterior torso,,female
19	ISIC_0000018_downsampled,30,posterior torso,,male
20	ISIC_0000019_downsampled,30,posterior torso,,female
21	ISIC_0000020_downsampled,25,anterior torso,,female
22	ISIC_0000021_downsampled,55,posterior torso,,female
23	ISIC_0000022_downsampled,55,lower extremity,,female
24	ISIC_0000023_downsampled,30,anterior torso,,female
25	ISIC_0000024_downsampled,45,posterior torso,,male
26	ISIC_0000025_downsampled,35,lower extremity,,female
27	ISIC_0000026_downsampled,30,lower extremity,,female

Fotoğrafları birbirinden ayırt etmek için 'image' sütünündan adları belirtilmiştir.Ayrıca hastalığa yakalanan ve yakalanmayan kişilerin bilgileri verilmiştir.Bazı kişilerin bilgileri verilmediğinden NULL olarak alınacaktır.Toplamda 25332 tane satır vardır.

Proje:

Spyder(Python 3.7) adlı açık kaynaklı platformdan geliştirilerek yapılmıştır.

2-Dataset okuma ve birleştirme:

İlk olarak kullanacağımız kütüphaneleri dahil edelim.

```
# %% Kütüphanelerim

import numpy as np#matematiksek işlemleriçin matrisler,
import pandas as pd#Csv ve text dosyalarını açmaya ve  içerisinde bulunan verileri okuyarak istenen sonuca  kolayca ulaşmak için kullanılmaktadır.
import matplotlib.pyplot as plt#grafik çizmek için
from sklearn.metrics import precision_recall_fscore_support as score#accuracy,f1score,... hesaplamak için
```

Veri setini okuyup birleştirme işlemi yapalım.

```
#%%Okuma ve metadataya ilk satıra göre birleştirme işlemi
metadata=pd.read_csv("ISIC_2019_Training_Metadata.csv")
groundata=pd.read_csv("ISIC_2019_Training_GroundTruth.csv") #df2

#out_f=pd.read_csv("out_f.csv")


df=pd.concat([metadata, groundata], axis=1, join='inner')
```

groundata - DataFrame

Index	image	MEL	NV	BCC	AK	BKL	DF	VASC	SCC	UNK
0	ISIC_0000000	0	1	0	0	0	0	0	0	0
1	ISIC_0000001	0	1	0	0	0	0	0	0	0
2	ISIC_0000002	1	0	0	0	0	0	0	0	0
3	ISIC_0000003	0	1	0	0	0	0	0	0	0
4	ISIC_0000004	1	0	0	0	0	0	0	0	0
5	ISIC_0000006	0	1	0	0	0	0	0	0	0
6	ISIC_0000007	0	1	0	0	0	0	0	0	0
7	ISIC_0000008	0	1	0	0	0	0	0	0	0
8	ISIC_0000009	0	1	0	0	0	0	0	0	0
9	ISIC_0000010	0	1	0	0	0	0	0	0	0
10	ISIC_0000011	0	1	0	0	0	0	0	0	0
11	ISIC_0000012	0	1	0	0	0	0	0	0	0
12	ISIC_0000013	1	0	0	0	0	0	0	0	0
13	ISIC_0000014	0	1	0	0	0	0	0	0	0
14	ISIC_0000015	0	1	0	0	0	0	0	0	0
15	ISIC_0000016	0	1	0	0	0	0	0	0	0
16	ISIC_0000017_downsampled	0	1	0	0	0	0	0	0	0
17	ISIC_0000018_downsampled	0	1	0	0	0	0	0	0	0
18	ISIC_0000019_downsampled	0	1	0	0	0	0	0	0	0
19	ISIC_0000020_downsampled	0	1	0	0	0	0	0	0	0

Format | Resize | Background color | Column min/max | Save and Close | Close

metadata - DataFrame

Index	image	age_approx	anatom_site_general	lesion_id	gender
0	ISIC_0000000	55	anterior torso	nan	female
1	ISIC_0000001	30	anterior torso	nan	female
2	ISIC_0000002	60	upper extremity	nan	female
3	ISIC_0000003	30	upper extremity	nan	male
4	ISIC_0000004	80	posterior torso	nan	male
5	ISIC_0000006	25	posterior torso	nan	female
6	ISIC_0000007	25	posterior torso	nan	female
7	ISIC_0000008	30	anterior torso	nan	female
8	ISIC_0000009	30	anterior torso	nan	female
9	ISIC_0000010	35	posterior torso	nan	female
10	ISIC_0000011	35	lower extremity	nan	female
11	ISIC_0000012	30	posterior torso	nan	male
12	ISIC_0000013	30	posterior torso	nan	female
13	ISIC_0000014	35	posterior torso	nan	male
14	ISIC_0000015	35	posterior torso	nan	male
15	ISIC_0000016	55	anterior torso	nan	female
16	ISIC_0000017_downsampled	50	posterior torso	nan	female
17	ISIC_0000018_downsampled	30	posterior torso	nan	male
18	ISIC_0000019_downsampled	30	posterior torso	nan	female
19	ISIC_0000020_downsampled	25	anterior torso	nan	female
20	ISIC_0000021_downsampled	55	posterior torso	nan	female

Format | Resize | Background color | Column min/max | Save and Close | Close

Birleştirmeden sonra

df - DataFrame

Index	image	ıge_appro	anatom_site_general	lesion_id	gender	image	MEL	NV	BCC	AK	BKL	DF	VASC	SCC	UNK
0	ISIC_0000000	55	anterior torso	nan	female	ISIC_0000000	0	1	0	0	0	0	0	0	0
1	ISIC_0000001	30	anterior torso	nan	female	ISIC_0000001	0	1	0	0	0	0	0	0	0
2	ISIC_0000002	60	upper extremity	nan	female	ISIC_0000002	1	0	0	0	0	0	0	0	0
3	ISIC_0000003	30	upper extremity	nan	male	ISIC_0000003	0	1	0	0	0	0	0	0	0
4	ISIC_0000004	80	posterior torso	nan	male	ISIC_0000004	1	0	0	0	0	0	0	0	0
5	ISIC_0000006	25	posterior torso	nan	female	ISIC_0000006	0	1	0	0	0	0	0	0	0
6	ISIC_0000007	25	posterior torso	nan	female	ISIC_0000007	0	1	0	0	0	0	0	0	0
7	ISIC_0000008	30	anterior torso	nan	female	ISIC_0000008	0	1	0	0	0	0	0	0	0
8	ISIC_0000009	30	anterior torso	nan	female	ISIC_0000009	0	1	0	0	0	0	0	0	0
9	ISIC_0000010	35	posterior torso	nan	female	ISIC_0000010	0	1	0	0	0	0	0	0	0
10	ISIC_0000011	35	lower extremity	nan	female	ISIC_0000011	0	1	0	0	0	0	0	0	0
11	ISIC_0000012	30	posterior torso	nan	male	ISIC_0000012	0	1	0	0	0	0	0	0	0
12	ISIC_0000013	30	posterior torso	nan	female	ISIC_0000013	1	0	0	0	0	0	0	0	0
13	ISIC_0000014	35	posterior torso	nan	male	ISIC_0000014	0	1	0	0	0	0	0	0	0

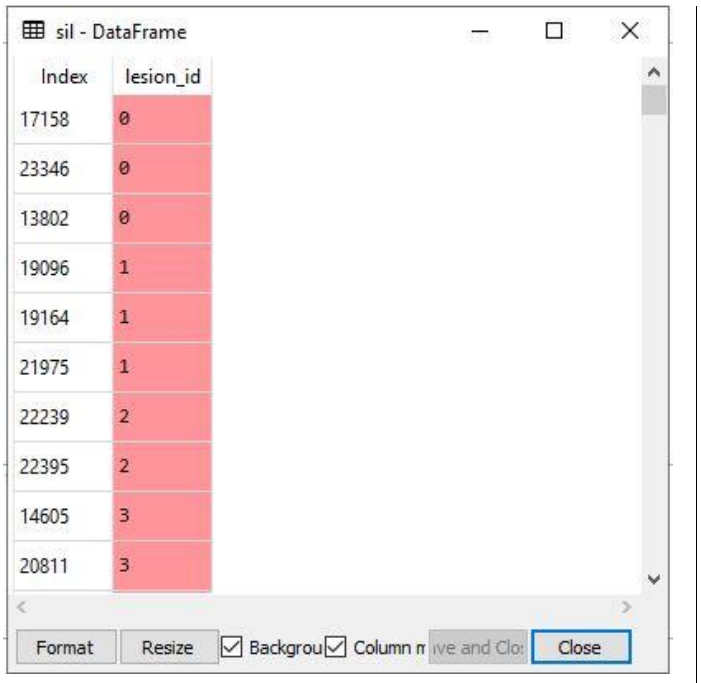
Format | Resize | Background color | Column min/max

3/4-NaN(Boş) değerleri veriseti'nden atma ve normalizasyon:

```
#%% NaN Degerleri silme
df.columns.values[5]="a"
df=df.dropna(how="any");#herhangi bir satırda NaN değeri varsa o satırı sil demek.
al=df.columns[[3,5,6,8,9,10,11,12,13,14]]

#names = df.columns.Tolist()

df_son = df.drop(df.columns[[3,5,6,8,9,10,11,12,13,14]], axis=1)

#Kalıcı hale getirmek için data.dropna(how="any",inplace=True) şeklinde yazmamız gerekir.


#%%
from sklearn.preprocessing import LabelEncoder
dt_frame_lession_id = df.loc[:,"lesion_id"]

sil=pd.DataFrame(dt_frame_lession_id).sort_values("lesion_id")


#print(sil[:1])
#print(len(dt_frame_lession_id)) //21311

for col in sil.columns:
    if sil[col].dtype == 'object':
        lab = LabelEncoder()
        encode = list(sil[col].values)
        lab.fit(encode)
        sil[col] = lab.transform(sil[col].values)
```

Object veri tipindeki değerleri sayısallaştırma

lesion_id
MSK4_0011169
MSK4_0011170
MSK4_0011171
MSK4_0011172
MSK4_0011173
MSK4_0010465
MSK4_0010119
MSK4_0011174
MSK4_0011175
MSK4_0011176
MSK4_0011177
MSK4_0011178
MSK4_0011179
MSK4_0011180



5-Grafiksel gösterim(“matplotlib.pyplot” BAR):

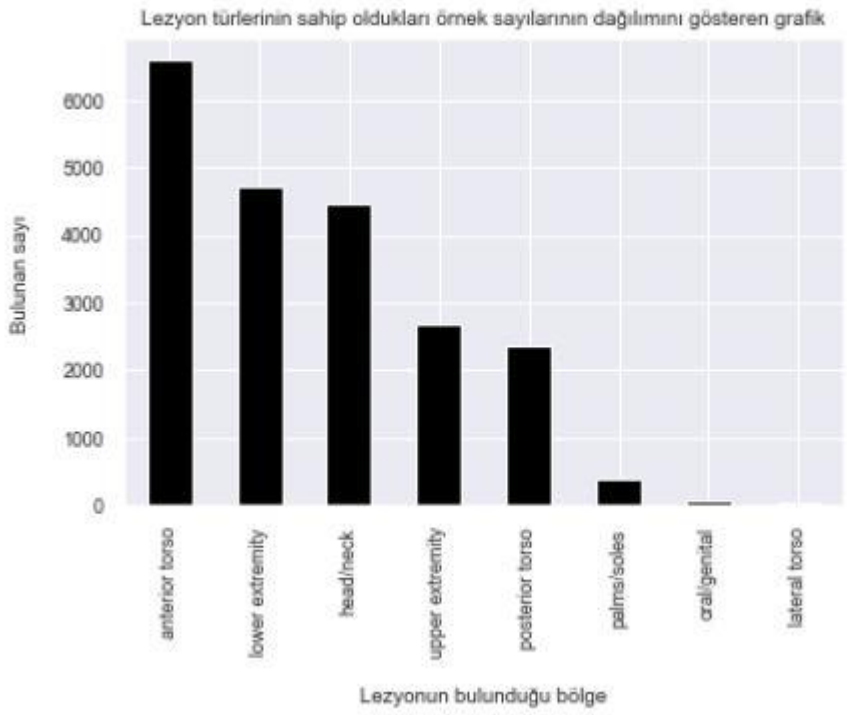
Matplotlib; 2 boyutlu grafikler hazırlamamızı sağlayan bir **Python** kütüphanesidir. Tanımı böyle olmasına rağmen 3 boyutlu görselleştirme de yapılabiliyor.

5.1- Verideki farklı lezyon türlerinin sahip oldukları örnek sayılarının dağılımını gösteren grafik;

import seaborn as sns // Kütüphane kısmına eklenmiştir.

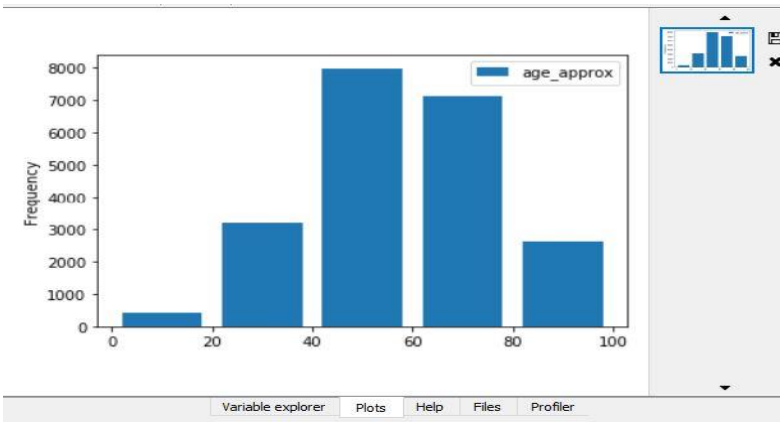
```
#%% -   Verideki farklı lezyon türlerinin sahip oldukları örnek sayılarının dağılımını gösteren grafik
sns.set(font_scale=0.8)
plt.title('Lezyon türlerinin sahip oldukları örnek sayılarının dağılımını gösteren grafik')
plt.xlabel("Lezyonun bulunduğu bölge", labelpad=14)
plt.ylabel("Bulunan sayı", labelpad=14)
df_son['anatom_site_general'].value_counts().plot(kind='bar',color=['black']);
```

Çıktısı



5.2- Verinin yaş dağılımını gösteren grafik

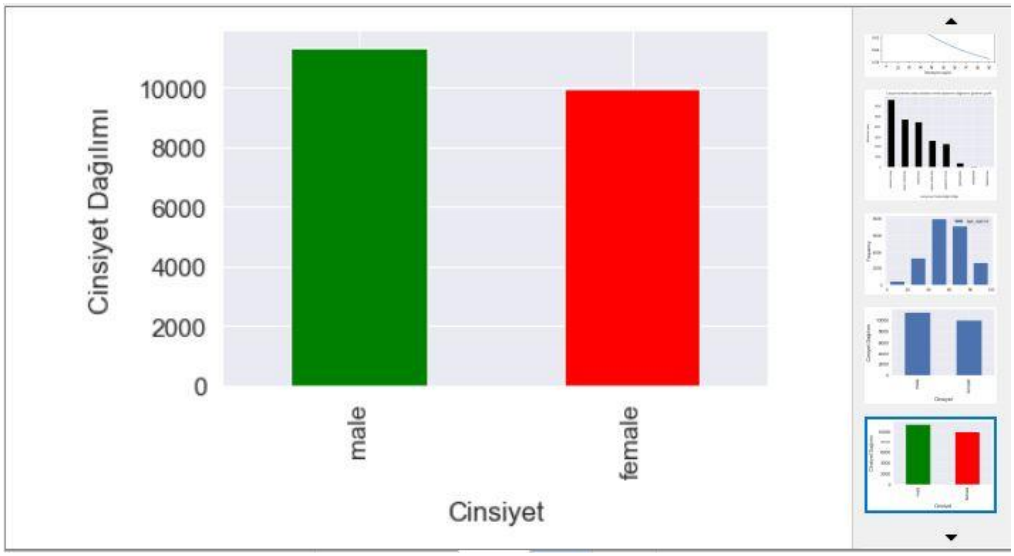
```
#%%-    Verinin yaş dağılımını gösteren grafik
sns.set(font_scale=1.2)
df_son[['age_approx']].plot(kind='hist',bins=[0,20,40,60,80,100],rwidth=0.7)

plt.show()#//Yaş dagılımı
```



5.3- Cinsiyete göre verilerin dağılımını gösteren bir grafik

```
#%% -   Cinsiyete göre verilerin dağılımını gösteren bir grafik
sns.set(font_scale=1.4)
plt.xlabel("Cinsiyet", labelpad=14)
plt.ylabel("Cinsiyet Dağılımı", labelpad=14)
df_son['gender'].value_counts().plot(kind='bar',color=['green', 'red'])
#plt.xticks(y_pos, bars));
```

Çıktısı:



EK-BONUS 5.4- Hasta olan ve olmayan sayı grafiği

```
        #%%Hasta olan ve olmayan sayı grafiği
bars = ('Hasta', 'Hasta Değil')
y_pos=np.arange(len(bars))
plt.title('Hasta olan ve olmayan sayı grafiği')
plt.xlabel("Hasta Olan ve Olmayan", labelpad=14)
plt.ylabel("Bulunan sayı", labelpad=14)

plt.bar(y_pos, df_son['NV'].value_counts(),width=0.6,color=['black', 'red'])
plt.xticks(y_pos, bars)
plt.show()
```

5.5- Sınıfı melanoma olup cinsiyet dağılımını gösteren grafik

```
#%% -   Sınıfı melanoma olup cinsiyet dağılımını gösteren grafik
melonamaolupcinsiyetler=df_son[df_son.NV==1]#MELONAMA OLANLARI AYIRIIYORUM
melonamaolupcinsiyetler[['age_approx']].plot(kind='hist',bins=[0,20,40,60,80,100],rwidth=0.7,color=['blue'])
```

melonamaolupcinsiyetler - DataFrame

Index	image	ıge_appro:	anatom_site_general	gender	NV
1459	ISIC_0012653_downsampled	50	posterior torso	female	1
1460	ISIC_0012654_downsampled	30	lower extremity	female	1
1461	ISIC_0012655_downsampled	35	upper extremity	female	1
1462	ISIC_0012656_downsampled	45	posterior torso	male	1
1463	ISIC_0012657_downsampled	20	upper extremity	female	1
1464	ISIC_0012658_downsampled	40	posterior torso	male	1
1465	ISIC_0012659_downsampled	30	posterior torso	female	1
1466	ISIC_0012660_downsampled	45	posterior torso	male	1
1468	ISIC_0012662_downsampled	75	posterior torso	male	1
1469	ISIC_0012663_downsampled	25	posterior torso	male	1
1470	ISIC_0012664_downsampled	40	upper extremity	female	1
1471	ISIC_0012665_downsampled	70	lower extremity	female	1
1473	ISIC_0012669_downsampled	55	lower extremity	male	1
1474	ISIC_0012670_downsampled	60	lower extremity	male	1
1475	ISIC_0012671_downsampled	60	lower extremity	male	1
1476	ISIC_0012672_downsampled	45	anterior torso	male	1
1477	ISIC_0012673_downsampled	45	anterior torso	male	1
1478	ISIC_0012674_downsampled	45	lower extremity	male	1
1479	ISIC_0012675_downsampled	45	lower extremity	male	1

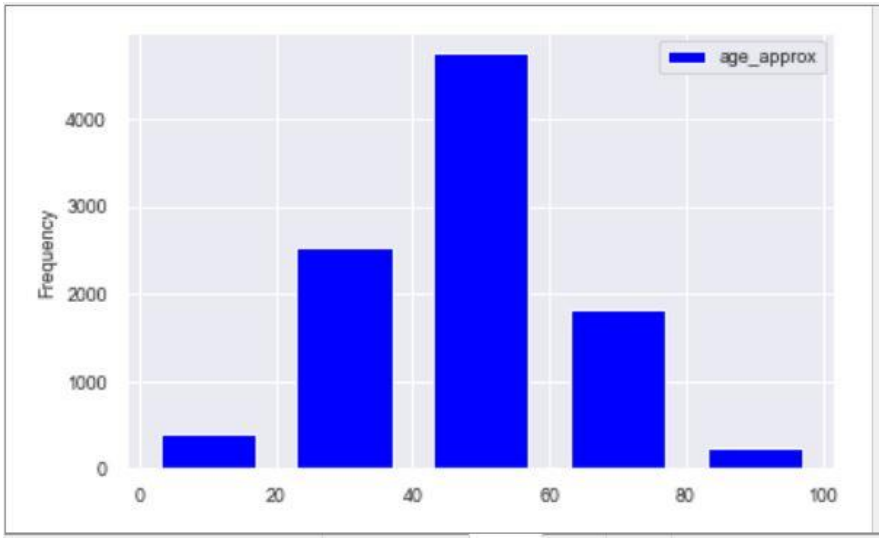
(NV) Melonoma olanlar için bölünmüştür.

Sadece NV=1 olanlar vardır.

Devam ediyor...

Format | Resize | Background color | Column min/max | Save and Close | Close

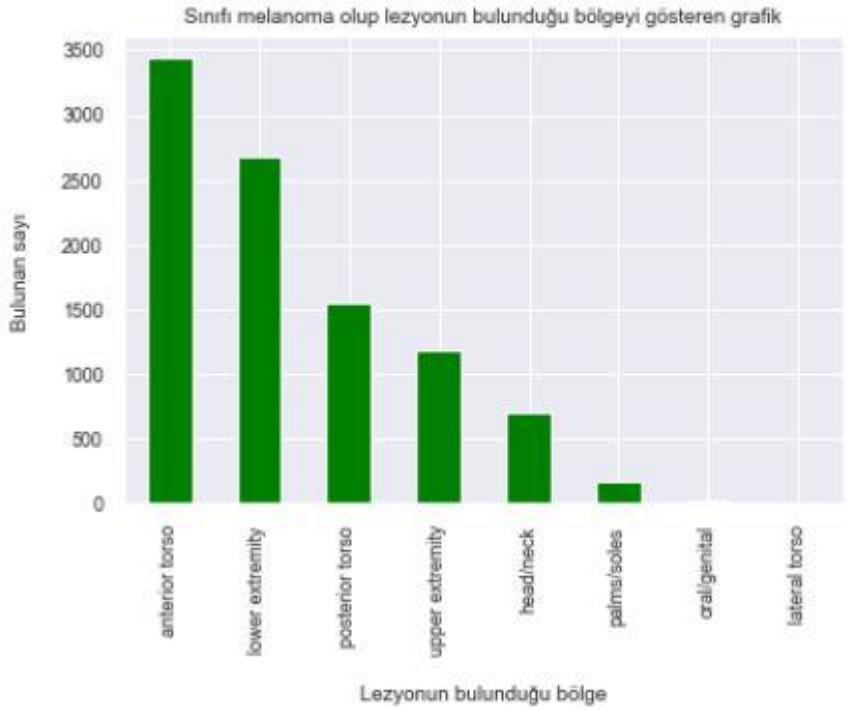
Çıktısı:



5.6- Sınıfı melanoma olup lezyonun bulunduğu bölgeyi gösteren grafik

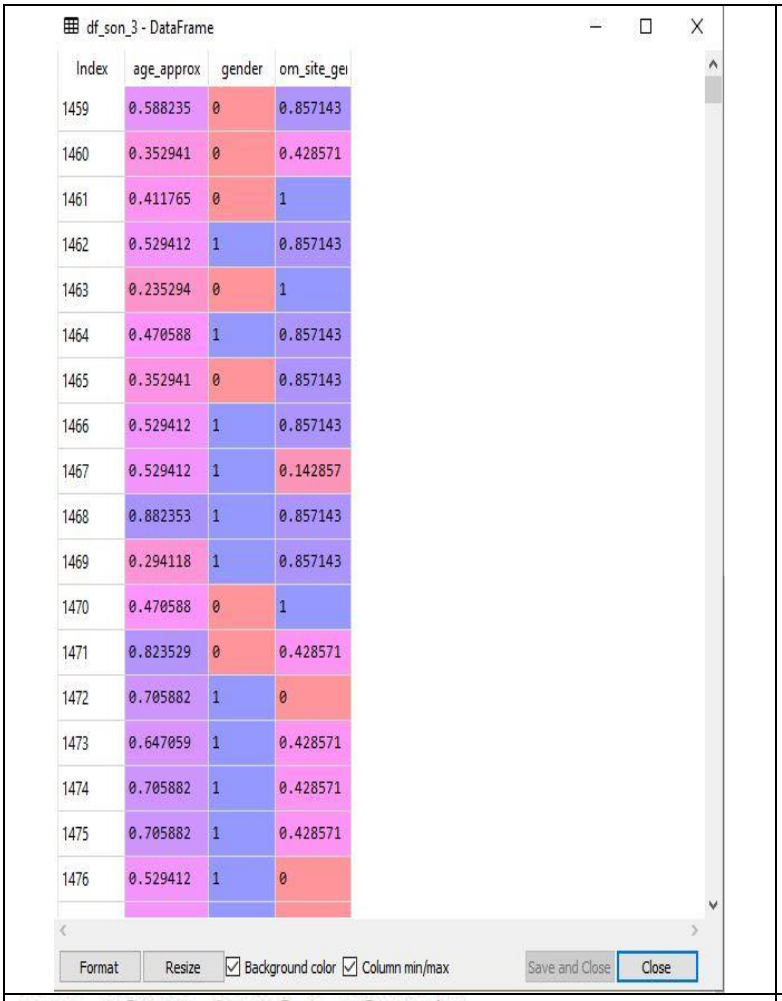
```
#%% -Sınıfı melanoma olup lezyonun bulunduğu bölgeyi gösteren grafik

sns.set(font_scale=0.8)
plt.title('Sınıfı melanoma olup lezyonun bulunduğu bölgeyi gösteren grafik')
plt.xlabel("Lezyonun bulunduğu bölge", labelpad=14)
plt.ylabel("Bulunan sayı", labelpad=14)
melonamaolupcinsiyetler['anatom_site_general'].value_counts().plot(kind='bar',color=['green']);
```

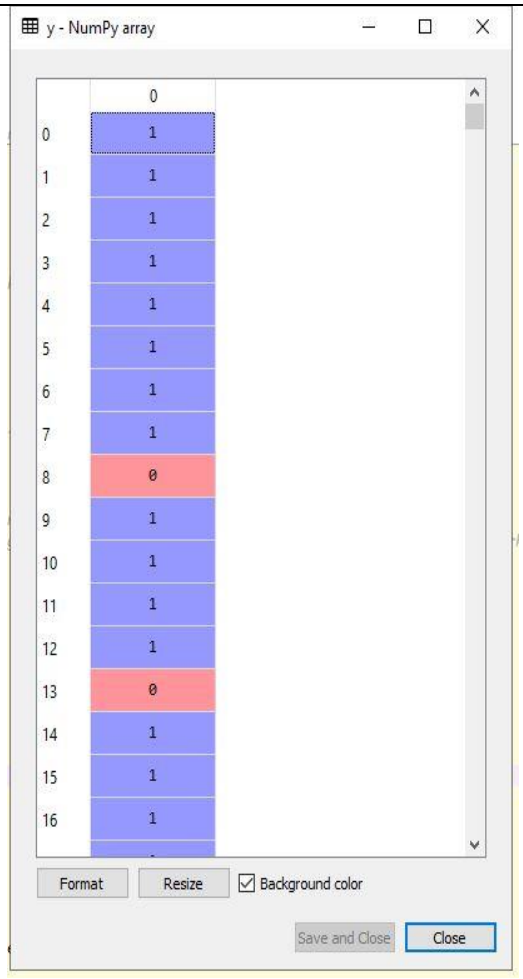
Çıktısı:



6- Dataset bölme(%20 Test-%80 Train) ve Normalisazyon

```
#%%DATASET BÖLME
# -------------------------------------
# -                     -             -
# -    %80 TRAİN        -  %20TEST    -
# -                     -             -
# -------------------------------------  OLSUN DİYORUM
#modelimi bu train ile eğiticeğim modelimi elde edegecegim,sonuçtan yüzde 20 siyle kontrol edicegim%%
from sklearn.model_selection import train_test_split
df_son.gender=[1 if each=="male" else 0  for each in df_son.gender ]
print(df_son.info())
# //sklearn model selection içinde traintestsplitle ayıran bir metodu var
df_son.drop(["image"],axis=1,inplace=True)

y=df_son.NV.values # datamın classların degerleri numpy array ceviriyordu//sınıflarının degerleri
#x=df_son.drop(["NV"],axis=1)

# x ve y ile al  0.2 yani yüzde %20 ile böl kalanı train olsun
# x(featureslarım) ve y(iyi huylu mu kotu huylu mu) x_train=%80 , x_test=%20,y_train=%80...
#bu metod 4 tane output döndürüyüor(random state=42  yazarıyorum ki 1 kere run ettiğimde aynı bölümlemeli yaparım yoksa her run ettiğimde farklı sonuclar cıkmasın diy
print(df_son.info())

df_anatom_site_general = df_son.loc[:,"anatom_site_general"]

sil=pd.DataFrame(df_anatom_site_general).sort_values("anatom_site_general")

for col in sil.columns:
    if sil[col].dtype == 'object':
        lab = LabelEncoder()
        encode = list(sil[col].values)
        lab.fit(encode)
        sil[col] = lab.transform(sil[col].values)
df_son.drop(["anatom_site_general","NV"],axis=1,inplace=True)
df_son_2=pd.concat([df_son, sil], axis=1, join='inner')
#Normalizasyon
#x = (x_data - np.min(x_data))/(np.max(x_data)-np.min(x_data))
df_son_3=(df_son_2 - np.min(df_son_2))/(np.max(df_son_2)-np.min(df_son_2))
#-------------------------------------
x_train,x_test,y_train,y_test=train_test_split(df_son_3,y,test_size=0.2,random_state=42)

print(df_son_2.info())
```





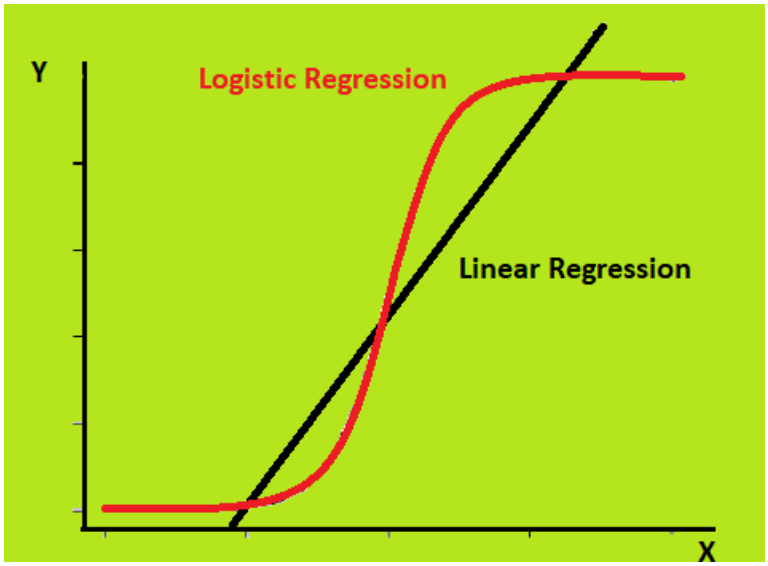

```
Data columns (total 3 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   age_approx            21311 non-null  float64
 1   gender                21311 non-null  int64
 2   anatom_site_general   21311 non-null  int32
dtypes: float64(1), int32(1), int64(1)
memory usage: 582.7 KB
```

```
#%%İşlem yaptırmak için transpozunu alıyorum-Alper Şahin ÖNER
x_train=x_train.T
x_test=x_test.T
y_train=y_train.T
y_test=y_test.T
```

5. Logistic Regression Sınıflandırma

Logistic Regression (Lojistik Regresyon) sınıflandırma işlemi yapmaya yarayan bir regresyon yöntemidir. Kategorik veya sayısal verilerin sınıflandırılmasında kullanılır. Bağımlı değişkenin yani sonucun sadece 2 farklı değer alabilmesi durumda çalışır. (Evet / Hayır, Erkek / Kadın, Şişman / Zayıf vs.)

Doğrusal sınıflandırma problemlerinde yaygın bir biçimde kullanılır. Bu sebeple Linear Regression ile çok benzemektedir.



Proje'de kullanımı

```
#%%-----------------------------------------------------------------------------------------
def initialize_weights_and_bias(dimension):
    w=np.full((dimension,1),0.01) #carpma işleminde 0 olmadığı için 0.01 olan matris yapıyorum
    b=0.0
    return w,b

#f(x)=1/1+e^-(z)

def sigmoid(z):
    y_head=1/(1+np.exp(-z))
    return y_head

#sigmoid(0)---->weight initialize yaptık
#%%
def forward_backward_propagation(w,b,x_train,y_train):
    # forward propagation
    z = np.dot(w.T,x_train) + b
    y_head = sigmoid(z)
    loss = -y_train*np.log(y_head)-(1-y_train)*np.log(1-y_head)
    cost = (np.sum(loss))/x_train.shape[1]      # x_train.shape[1]  is for scaling

    # backward propagation
    derivative_weight = (np.dot(x_train,((y_head-y_train).T)))/x_train.shape[1] # x_train.shape[1]  is for scaling
    derivative_bias = np.sum(y_head-y_train)/x_train.shape[1]                 # x_train.shape[1]  is for scaling
    gradients = {"derivative_weight": derivative_weight, "derivative_bias": derivative_bias}

    return cost,gradients
```

```
#%% updating (learning) parametres
#weight ve bias iterasyon sayısı kadar guncelleme yapıcam
#(ileri git geri git türev al integral al şeklinde güüüncelleme yaptıracagım)
    #modelimi güncelledigimde train ediyorum. data mın feature larımı (kanser hücrem kötü huylumu iyi huylumu)
def update(w, b, x_train, y_train, learning_rate,number_of_iterarion):
    cost_list = []
    cost_list2 = []
    index = []

    # updating(learning) parameters is number_of_iterarion times
    for i in range(number_of_iterarion):
        # make forward and backward propagation and find cost and gradients
        cost,gradients = forward_backward_propagation(w,b,x_train,y_train) # sigmoid func gelen degerler
        cost_list.append(cost)
        # Güncelleme
        w = w - learning_rate * gradients["derivative_weight"]
        b = b - learning_rate * gradients["derivative_bias"]
        if i % 10 == 0:
            cost_list2.append(cost)
            index.append(i)
            print ("İterasyon sonrası maliyet %i: %f" %(i, cost))

    # we update(learn) parameters weights and bias
    parameters = {"weight": w,"bias": b} # sözlügün icinde depoluyorum
    plt.plot(index,cost_list2)
    plt.xticks(index,rotation='vertical')
    plt.xlabel("İterasyon sayısı")
    plt.ylabel("Maliyet")
    plt.show()
    return parameters, gradients, cost_list
```

```
#%%  # prediction
#(tümorlu datam iyi huylumu kotu huylumu kontrol edicem)
def predict(w,b,x_test):
    # x_test is a input for forward propagation
    z = sigmoid(np.dot(w.T,x_test)+b)
    Y_prediction = np.zeros((1,x_test.shape[1]))
    # if z is bigger than 0.5, our prediction is sign one (y_head=1),
    # if z is smaller than 0.5, our prediction is sign zero (y_head=0),
    for i in range(z.shape[1]):
        if z[0,i]<= 0.5:
            Y_prediction[0,i] = 0
        else:
            Y_prediction[0,i] = 1

    return Y_prediction
```

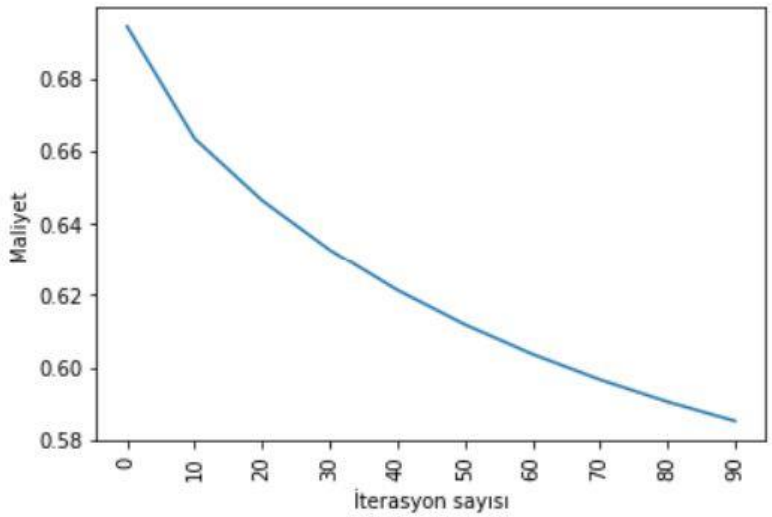
```python
# %% logistic_regression
def logistic_regression(x_train, y_train, x_test, y_test, learning_rate ,  num_iterations):
    # initialize
    dimension =  x_train.shape[0]  # that is 30
    w,b = initialize_weights_and_bias(dimension)
    # do not change learning rate
    parameters, gradients, cost_list = update(w, b, x_train, y_train, learning_rate,num_iterations)

    y_prediction_test = predict(parameters["weight"],parameters["bias"],x_test)

    precision, recall, fscore, support = score(y_test.T, y_prediction_test.T)
    print('---------------------------')
    print('precision: {}'.format(precision))
    print('recall: {}'.format(recall))
    print('fscore: {}'.format(fscore))
    print('support: {}'.format(support))
    print("Lineer Regresyon accuracy: {} %".format(100 - np.mean(np.abs(y_prediction_test - y_test)) * 100))
    print('---------------------------')
logistic_regression(x_train, y_train, x_test, y_test,learning_rate = 1, num_iterations = 100)
```

#sklearn kütüphanesinin içindeki LogisticRegression sınıfını kullanarak yapmak.

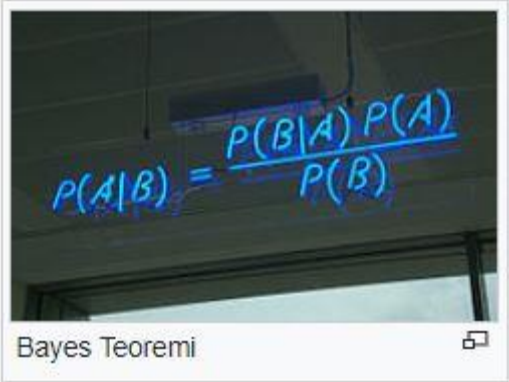
```python
#%% sklearn with LR
#daha kolay kod yazar bunu çözücez
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
lr.fit(x_train.T,y_train.T)
print("Logistic Regresyontest accuracy (sklearn) {}".format(lr.score(x_test.T,y_test.T)*100))
print('---------------------------')
```

Çıktısı:

```
İterasyon sonrası maliyet 0: 0.694393
İterasyon sonrası maliyet 10: 0.663477
İterasyon sonrası maliyet 20: 0.646435
İterasyon sonrası maliyet 30: 0.632738
İterasyon sonrası maliyet 40: 0.621411
İterasyon sonrası maliyet 50: 0.611864
İterasyon sonrası maliyet 60: 0.603712
İterasyon sonrası maliyet 70: 0.596687
İterasyon sonrası maliyet 80: 0.590592
İterasyon sonrası maliyet 90: 0.585276
---------------------------
precision: [0.71877395 0.72171809]
recall: [0.80308219 0.61909704]
fscore: [0.7585928  0.66648045]
support: [2336 1927]
Lineer Regresyon accuracy: 71.99155524278677 %
---------------------------
Logistic Regresyontest accuracy (sklearn) 72.1323011963406
---------------------------
```



6. Naive Bayes Sınıflandırma



Bayes Teoremi

P(*A*|*B*) ; B olayı gerçekleştiği durumda A olayının meydana gelme olasılığıdır (bakınız koşullu olasılık)

P(*B*|*A*) ; A olayı gerçekleştiği durumda B olayının meydana gelme olasılığıdır

P(A) ve P(B) ; A ve B olaylarının önsel olasılıklarıdır.

Naive Bayes Sınıflandırıcı adını İngiliz matematikçi Thomas Bayes'ten (yak. 1701 - 7 Nisan 1761) alır.

Naive Bayes Sınıflandırıcı Örüntü tanıma problemine ilk bakışta oldukça kısıtlayıcı görülen bir önerme ile kullanılabilen olasılıkcı bir yaklaşımdır. Bu önerme örüntü tanımada kullanılacak her bir tanımlayıcı öznitelik ya da parametrenin istatistik açıdan bağımsız olması gerekliliğidir. Her ne kadar bu önerme Naive Bayes sınıflandırıcının kullanım alanını kısıtlasa da, genelde istatistik bağımsızlık koşulu esnetilerek kullanıldığında da daha karmaşık yapay sinir ağları gibi metotlarla karşılaştırabilir sonuçlar vermektedir.

Bir Naive Bayes sınıflandırıcı, her özniteliğin birbirinden koşulsal bağımsız olduğu ve öğrenilmek istenen kavramın tüm bu özniteliklere koşulsal bağlı olduğu bir Bayes ağı olarak da düşünülebilir.

Projede kullanımı

```
#%% # %% Naive bayes
from sklearn.naive_bayes import GaussianNB
nb = GaussianNB()
nb.fit(x_train.T,y_train.T)
print("Naive Bayes Algoritması doğruluk: ",nb.score(x_test.T,y_test.T))
print('----------------------------')
```

Çıktısı:

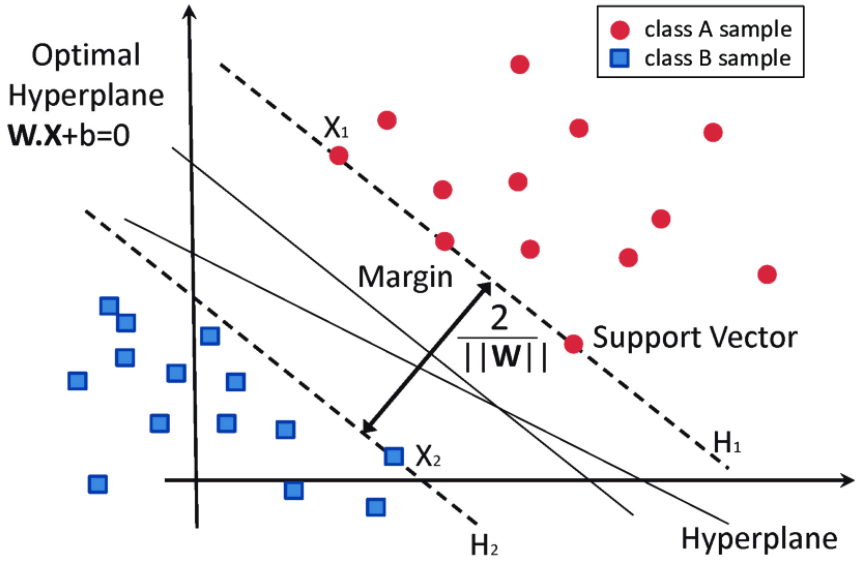
```
----------------------------
Naive Bayes Algoritması doğruluk:  0.7206192821956369
----------------------------
```

6. Support Vector Machine(SVM)

Destek vektör makinesi (kısaca **DVM**), eğitim verilerindeki herhangi bir noktadan en uzak olan iki sınıf arasında bir karar sınırı bulan vektör uzayı tabanlı makine öğrenme yöntemi olarak tanımlanabilir.

Destek Vektör Makineleri, temel olarak iki sınıfa ait verileri birbirinden en uygun şekilde ayırmak için kullanılır. Bunun için karar sınırları yada diğer bir ifadeyle hiper düzlemler belirlenir.DVM'ler günümüzde yüz tanıma sistemlerinden, ses analizine kadar birçok sınıflandırma probleminde kullanılmaktadırlar.
Avantajları:

- Yüksek boyutlu uzaylarda etkilidirler.
- Boyut sayısının, örnemlem sayısından fazla olduğu durumlarda etkilidirler.
- Karar fonksiyonunda bir takım eğitim noktaları kullanılır ("support vectors"). Dolayısıyla bellek verimli bir şekilde kullanılmış olur.
- Çok yönlü: Karar fonksiyonu için çok farklı çekirdek fonksiyonları ("kernel functions") kullanılabilmektedir.



Projede kullanımı

```
# %% SVM
from sklearn.svm import SVC
svm = SVC(random_state = 1)
svm.fit(x_train.T,y_train.T)
print('----------------------------')
print("SVM ile doğruluk: ",svm.score(x_test.T,y_test.T))
print('----------------------------')
```

Çıktısı:

```
----------------------------
SVM ile doğruluk:  0.7262491203377903
----------------------------
```

KAYNAKÇA

https://medium.com/@k.ulgen90/makine-öğrenimi-bölüm-4-destek-vektör-makineleri-2f8010824054
https://tr.wikipedia.org/wiki/Destek_vektör_makinesi
https://tr.wikipedia.org/wiki/Naive_Bayes_sınıflandırıcı
https://medium.com/@ekrem.hatipoglu/machine-learning-classification-logistic-regression-part-8-b77d2a61aae1
https://www.udemy.com/course/machine-learning-ve-python-adan-zye-makine-ogrenmesi-4/
https://medium.com/datarunner/python-ile-veri-görselleştirme-matplotlib-kütüphanesi-2-8d1cffef75e8
https://scikit-learn.org/stable/auto_examples/applications/plot_face_recognition.html#sphx-glr-auto-examples-applications-plot-face-recognition-py
https://stackoverflow.com/questions/33275461/specificity-in-scikit-learn
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
https://scikit-learn.org/stable/modules/model_evaluation.html
https://ertugruldeniz.com/python-pandas-nedir-ve-nasil-kullanilir-141
https://medium.com/bili%C5%9Fim-hareketi/veri-bilimi-i%CC%87%C3%A7in-temel-python-k%C3%BCt%C3%BCphaneleri-2-pandas-dcc12ae01b7d
https://gelecegiyazanlar.turkcell.com.tr/blog/python-kutuphanesi-pandas
https://medium.com/@aykutuludag/numpy-k%C3%BCt%C3%BCphanesi-75667e99c42a