Proje Tanımı ve Teknik Yol Haritası

Proje Amacı:

Bu proje, pnömoninin(akciğer enfeksiyonu) erken teşhisi amacıyla, tıbbi görüntü analizi yöntemlerini kullanarak, hastalık belirtilerini otomatik olarak tespit edebilen bir sistem geliştirmeyi hedefliyoruz. Erken teşhis sayesinde, hastaların tedaviye erken başlamaları sağlanarak, hayat kurtarıcı sonuçlara ulaşılması amaçlanmaktadır. Proje ilerleyen aşamalarda yalnızca pnömoni değil, benzer klinik belirtiler gösteren diğer akciğer hastalıklarına da uyarlanabilir.

Kullanılacak Veri Setleri:

Projede, özellikle akciğer röntgen görüntülerine odaklanılacaktır. Kullanılabilecek temel veri setleri arasında:

ChestX-ray14: NIH tarafından sağlanan, 100.000'den fazla göğüs röntgenini içeren, 14 farklı akciğer hastalığını kapsayan geniş veri seti.

COVID-19 Image Data Collection: COVID-19 pnömonisi teşhisi için kullanılan çeşitli kaynaklardan derlenmiş göğüs röntgeni ve CT görüntülerini içeren veri setleri.

RSNA Pneumonia Detection Challenge Dataset: Pnömoni teşhisi için özel olarak oluşturulmuş röntgen görüntülerini içermektedir.

CheXpert Dataset: Beş farklı akciğer hastalığını içeren büyük ölçekli bir veri setidir.

Teknik Yol Haritası:

Veri Toplama ve Ön İşleme:

Veri Analizi: Kullanılacak veri setlerinin incelenmesi, istatistiksel analizler yapılarak veri dağılımının ve örnek sayısının belirlenmesi.

Görüntü İşleme:

Gürültüden Kurtarma (Denoising): Görüntülerdeki gürültülerin giderilmesi için filtreleme yöntemleri uygulanacak.

Boyutlandırma (Resizing): Tüm görüntüler standart boyutlara yeniden boyutlandırılarak modelin girişine uygun hale getirilecek.

Normalizasyon : Görüntü parlaklığı, kontrastı ve diğer özellikler normalize edilecek.

Model Geliştirme:

Klasik Makine Öğrenmesi Yaklaşımı:

İlk aşamada, geleneksel makine öğrenmesi algoritmaları (hangi modelin kullanılacağını makaleler ile belirleyeceğiz) kullanılarak, önceden çıkarılan özellikler (örneğin, histogram, kenar tespiti) üzerinden sınıflandırma modelleri geliştirilecek.

Transfer Öğrenimi Tabanlı CNN Yaklaşımı:

Klasik yöntemlerle elde edilen sonuçların ardından, derin öğrenmeye geçiş yapılacak.

Transfer öğrenimi kullanılarak, önceden büyük veri setleri üzerinde eğitilmiş bir konvolüsyonel sinir ağı (CNN) seçilecek ve akciğer röntgen görüntülerine ince ayar uygulanacak.

Bu yaklaşım, modelin klinik ortamlarda yüksek performans göstermesi ve genel geçer bir özellik çıkarımı yapabilmesi için tercih edilecektir.

Değerlendirme ve Genişletme:

Performans Ölçütleri: Modelin metriklerle değerlendirilmesi yapılacak.

Gelecekteki Genişletmeler: Eğer başarılı olunursa, model farklı akciğer hastalıklarını da içerecek şekilde genişletilebilir ve sağlık alanında daha geniş bir kullanım alanı bulabilir.

Proje Özeti:

Bu proje, pnömoninin erken teşhisi için geniş çaplı akciğer röntgeni veri setleri kullanılarak, öncelikle klasik makine öğrenmesi teknikleriyle temel model geliştirmeyi, sonrasında ise transfer öğrenimi tabanlı CNN modelleri ile görüntü analizi, gürültü giderme, boyutlandırma ve normalizasyon işlemlerini entegre eden kapsamlı bir sistem geliştirmeyi hedeflemektedir. Amacımız, erken teşhis sayesinde hastaların hayatlarını kurtarmak ve ilerleyen aşamalarda sistemimizi diğer akciğer hastalıklarına da genişleterek toplumsal sağlık hizmetlerine katkı sağlamaktır.

*Projeyi belirlediğimiz geçtiğimiz 2 hafta boyunca öğrendiklerimizi yansıtmak adına bu dökümantasyonu hazırladık

1.Hafta : Python Temelleri ve numpy, pandas, matplotlib kütüphanelerinin kullanımı

2.Hafta : Temel Veri Analizi , Veri Okuryazarlığı, Kaggle Platformu

Giriş

Python'un veri analizi ve görselleştirme için yaygın olarak kullanılan NumPy, Pandas ve Matplotlib kütüphaneleri kullanılmıştır.

Python - Kullanılan Kütüphaneler ve Amaçları

Bu süreçte kullanılan kütüphaneler, verilerin doğru bir şekilde işlenmesini, analiz edilmesini ve görselleştirilmesini sağlar.

NumPy

- NumPy, büyük boyutlu diziler ve matrisler üzerinde hızlı hesaplamalar yapmayı sağlayan bir kütüphanedir.

Pandas

- Pandas, veri manipülasyonu ve analizi için güçlü bir kütüphanedir.
- Veri çerçeveleri (DataFrame) kullanarak CSV, Excel ve SQL gibi formatlardaki veriler üzerinde işlem yapılmasını sağlar.

Matplotlib

- Matplotlib, veri görselleştirmesi yapmak için kullanılan bir kütüphanedir.
- Grafikler ve dağılım diyagramları oluşturulmasını sağlar.

Veri Türleri ve Kullanımları

Numeric (Sayısal Veriler)

- Sürekli veya kesikli değerlerdir.
- İstatistiksel hesaplamalar ve görselleştirmeler için kullanılır.

Categoric (Kategorik Veriler)

- Belirli gruplara ayrılmış veriler olup genellikle metin veya sayısal değerler şeklindedir.

Num & Cat (Sayısal ve Kategorik Verilerin Birleşimi)

- Makine öğrenimi modellerinde sayısal ve kategorik veriler birlikte kullanılabilir.
- Kategorik veriler, one-hot encoding gibi tekniklerle sayısal verilere dönüştürülür.

Maps (Harita Verileri)

- Coğrafi konum bilgilerini içerir ve harita tabanlı görselleştirmelerde kullanılır.

Network (Ağ Verileri)

- Düğümler ve bağlantılar arasındaki ilişkileri gösterir. Sosyal ağlar ve internet trafiği gibi analizlerde kullanılır.

Time Series (Zaman Serisi Verileri)

- Zaman içinde değişen verilerdir.
- Finansal analiz ve hava durumu tahminlerinde kullanılır.

Yapılan İşlemlerden Ekran Görüntüleri:

- Olimpiyat sporcularının veriseti kullanılacaktır.

```
# Kütüphaneleri import edelim
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from collections import Counter

# Python uyarıları kapatalım
import warnings
warnings.filterwarnings("ignore")
```

```
#veriyi içeri atalım
veri = pd.read_csv("veriseti_20220203_olimpiyatlar.csv")
veri.head()
```

	ID	Name	Gender	Age	Height	Weight	Team	NOC	Games	Year	Season	City	Sport	Event	Medal
0	1	A Dijiang	M	24.0	180.0	80.0	China	CHN	1992 Summer	1992	Summer	Barcelona	Basketball	Basketball Men's Basketball	NaN
1	2	A Lamusi	M	23.0	170.0	60.0	China	CHN	2012 Summer	2012	Summer	London	Judo	Judo Men's Extra-Lightweight	NaN
2	3	Gunnar Nielsen Aaby	M	24.0	NaN	NaN	Denmark	DEN	1920 Summer	1920	Summer	Antwerpen	Football	Football Men's Football	NaN
3	4	Edgar Lindenau Aabye	M	34.0	NaN	NaN	Denmark/Sweden	DEN	1900 Summer	1900	Summer	Paris	Tug-Of-War	Tug-Of-War Men's Tug-Of-War	Gold
4	5	Christine Jacoba Aaftink	F	21.0	185.0	82.0	Netherlands	NED	1988 Winter	1988	Winter	Calgary	Speed Skating	Speed Skating Women's 500 metres	NaN

- Pandas kütüphanesi kullanarak, CSV (comma separated values) değerlerini listeledik.

Veri Hakkında Bilgi

- veriyi tanımak

```
veri.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 271116 entries, 0 to 271115
Data columns (total 15 columns):
 #   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   ID      271116 non-null  int64
 1   Name    271116 non-null  object
 2   Gender  271116 non-null  object
 3   Age     261642 non-null  float64
 4   Height  210945 non-null  float64
 5   Weight  208241 non-null  float64
 6   Team    271116 non-null  object
 7   NOC     271116 non-null  object
 8   Games   271116 non-null  object
 9   Year    271116 non-null  int64
 10  Season  271116 non-null  object
 11  City    271116 non-null  object
 12  Sport   271116 non-null  object
 13  Event   271116 non-null  object
 14  Medal   39783 non-null   object
dtypes: float64(3), int64(2), object(10)
memory usage: 31.0+ MB
```

- 14 sütun ve adlarını öğrenirken, her sütundaki veri miktari ve veri tipini görüntüledik. Eksik veriler içerdiğini keşfedildi.

Verilerin Temizlenmesi

- Sütun isimlerinin düzenlenmesi
- Yararsız verilerin çıkartılması ve düzenlenmesi
- Kayıp veri sorunu

Sütun isimlerinin düzenlenmesi

- Verilerin içerisinde bulunan sütun isimlerini inceleyeceğiz ve bu isimleri türkçeye çevireceğiz.

```
veri.columns
```

```
Index(['ID', 'Name', 'Gender', 'Age', 'Height', 'Weight', 'Team', 'NOC',
       'Games', 'Year', 'Season', 'City', 'Sport', 'Event', 'Medal'],
      dtype='object')
```

```
# sütun isimlerini değiştirelim
veri.rename(columns={'ID'    :   'id',
                     'Name'    :   'isim',
                     'Gender'    :   'cinsiyet',
                     'Age'    :   'yas',
                     'Height'    :   'boy',
                     'Weight'    :   'kilo',
                     'Team'    :   'takim',
                     'NOC'    :   'uok',
                     'Games'    :   'oyunlar',
                     'Year'    :   'yil',
                     'Season'    :   'sezon',
                     'City'    :   'sehir',
                     'Sport'    :   'spor',
                     'Event'    :   'etkinlik',
                     'Medal'    :   'madalya'}, inplace = True)
# inplace, üzerine yaz demek.
```

```
veri.head(2)
```

	id	isim	cinsiyet	yas	boy	kilo	takim	uok	oyunlar	yil	sezon	sehir	spor	etkinlik	madalya
0	1	A Dijiang	M	24.0	180.0	80.0	China	CHN	1992 Summer	1992	Summer	Barcelona	Basketball	Basketball Men's Basketball	NaN
1	2	A Lamusi	M	23.0	170.0	60.0	China	CHN	2012 Summer	2012	Summer	London	Judo	Judo Men's Extra-Lightweight	NaN

- Sütun adlarını Türkçeleştirildi.

Yararsız Verilerin Çıkartılması ve Düzenlenmesi

- id yararsızdır.
- oyunlar = yıl + sezon old. için yararsızdır.

```
# drop fonk. ile id ve oyunlar ı çıkartalım.
veri = veri.drop(["id","oyunlar"], axis = 1)
# axis=1 sütun, axis=0 satır
```

```
veri.head(2)
```

	isim	cinsiyet	yas	boy	kilo	takim	uok	yil	sezon	sehir	spor	etkinlik	madalya
0	A Dijiang	M	24.0	180.0	80.0	China	CHN	1992	Summer	Barcelona	Basketball	Basketball Men's Basketball	NaN
1	A Lamusi	M	23.0	170.0	60.0	China	CHN	2012	Summer	London	Judo	Judo Men's Extra-Lightweight	NaN

- oyunlar(yil + sezon) ve id sütunlarının herhangi bir bilgi vermediği tespit edilip çıkartıldı. Bu sayede verisetinin boyutu düşürüldü.

Verilere İlişkin Sorunlar

Hatalı / Kirli Veriler

- Kaydedilmemiş / Girilmemiş Veriler

    meslek =

- Hatalı Girilmiş Veriler

    maaş = '-10'

- Tutarsız Nitelik İsimleri

    Bir yerde '*ad*', başka yerde '*isim*'

    Bir yerde '*soyadı*', başka yerde '*soyismi*'

    Bir yerde 'malzeme referans no', başka yerde 'malzeme kayıt no'

- Tutarsız Nitelik Değerleri

    Bir yerde 'yaş=30-40 arası', başka yerde 'doğum tarihi = 01.02.1970'

Hatalı / Kirli Verilerin Nedenleri

- Eksik veri kayıtlarının nedenleri

    Veri toplandığı sırada bir nitelik değerinin elde edilememesi, bilinmemesi

    Veri toplandığı sırada bazı niteliklerin gerekliliğinin görülememesi

    İnsan, yazılım ya da donanım problemleri

- Gürültülü (hatalı) veri kayıtlarının nedenleri

    Hatalı veri toplama gereçleri

    İnsan, yazılım ya da donanım problemleri

    Veri iletimi sırasında problemler

- Tutarsız veri kayıtlarının nedenleri

    Verinin farklı veri kaynaklarında tutulması

    İşlevsel bağımlılık kurallarına uyulmaması

Eksik Veriyi Tamamlama

- Eksik nitelik değerleri olan kayıtların kullanılmaması / atılması. Fakat bu durum elimizdeki değerli veriyi kaybetmemiz ile sonuçlanır. Aşağıdaki yaptığımız örnekte madalyası olmayan sporcuları sildik.

[32]:
```
#madalya alamayan sporcuları veri setinden çıkaracağız
madalya_degiskeni = veri["madalya"]
pd.isnull(madalya_degiskeni).sum()
```

[32]:
```
np.int64(231333)
```

[33]:
```
madalya_degiskeni_filtresi = ~pd.isnull(madalya_degiskeni)
veri = veri[madalya_degiskeni_filtresi]
veri.head(5)
```

[33]:

	isim	cinsiyet	yas	boy	kilo	takim	uok	yil	sezon	sehir	spor	etkinlik	madalya
3	Edgar Lindenau Aabye	M	34.0	182.48	95.62	Denmark/Sweden	DEN	1900	Summer	Paris	Tug-Of-War	Tug-Of-War Men's Tug-Of-War	Gold
37	Arvo Ossian Aaltonen	M	30.0	182.01	76.69	Finland	FIN	1920	Summer	Antwerpen	Swimming	Swimming Men's 200 metres Breaststroke	Bronze
38	Arvo Ossian Aaltonen	M	30.0	177.00	75.00	Finland	FIN	1920	Summer	Antwerpen	Swimming	Swimming Men's 400 metres Breaststroke	Bronze
40	Juhamatti Tapio Aaltonen	M	28.0	184.00	85.00	Finland	FIN	2014	Winter	Sochi	Ice Hockey	Ice Hockey Men's Ice Hockey	Bronze
41	Paavo Johannes Aaltonen	M	28.0	175.00	64.00	Finland	FIN	1948	Summer	London	Gymnastics	Gymnastics Men's Individual All-Around	Bronze

- Eksik nitelik değerlerinin, aynı sınıfa ait kayıtların nitelik değerlerinin ortalaması ile doldurulması. Tüm veri setindeki nitelik ortalaması değil örneğin aynı etkinlikte bulunan sporcuların boy ve kilo ortalaması ile değerler dolduruluyor.

[30]:
```
#her bir etkinliği iteratif olarak dolaş
#etkinlik özelinde boy ve kilo ort hesapla
#etkinlik özelinde kayı boy ve kilo değerlerini etkinlik ortalamalarına eşitle
veri_gecici = veri.copy() #gerçek veriyi bozmamak için bir kopyasını oluşturdum
boy_kilo_liste = ["boy", "kilo"]

for e in essiz_etkinlik : #liste içerisinde dolaş

    #etkinlik filtresi oluşturalım
    etkinlik_filtre = veri_gecici.etkinlik == e
    #veriyi etkinliğe göre filtreleyelim
    veri_filtreli = veri_gecici[etkinlik_filtre]

    for s in boy_kilo_liste:
        ortalama = np.round(np.mean(veri_filtreli[s]),2)
        if ~np.isnan(ortalama): #eğer etkinlik özelinde ortalama varsa
            veri_filtreli[s] = veri_filtreli[s].fillna(ortalama)
        else:
            tum_veri_ortalamasi = np.round(np.mean(veri[s]),2)
            veri_filtreli[s] = veri_filtreli[s].fillna(tum_veri_ortalamasi)

    #etkinlik özelinde kayıp değerleri doldurulmul olan veriyi, veri_gecici ye eşitleyelim
    veri_gecici[etkinlik_filtre] = veri_filtreli

#kayıp değerleri giderilmiş olan geçici veriyi gerçek veriye eşitle
veri = veri_gecici.copy()
veri.info()
```

- Eksik nitelik değerlerinin manuel olarak doldurulması. Bu yöntem kendi elimiz ile doldurmamızı kapsıyor.

- Eksik nitelik değerleri için global bir değişken kullanılması (Null, Bilinmiyor, vb.)

- Eksik nitelik değerlerinin, o niteliğin ortalama değeri ile doldurulması yani tüm veri setindeki nitelik ortalamasından bahsediyoruz.

- Eksik nitelik değerlerinin, olasılığı en fazla olan nitelik değeriyle doldurulması. eksik değerler en sık görülen (mod) değerle doldurulur. Özellikle kategorik veriler için kullanışlıdır. Örneğin şehirlerde eksik varsa ve en çok kullanılan şehir İstanbul ise eksikler İstanbul ile kapatılır.

Gürültülü Veri Nasıl Düzeltilir ?

- Bölütleme (Segmentation)

Verinin sıralanması, eşit genişlik veya eşit derinlik ile bölünmesi

Veri: 8, 4, 21, 15, 21, 25, 24, 34, 28

Sıralı Veri: 4, 8, 15, 21, 21, 24, 25, 28, 34

Eşit Genişlik Yaklaşımı: Bölme sayısının belirlenmesi ve verinin eşit aralıklarla bölünmesi

Eşit Derinlik Yaklaşımı: Her bölmede eşit sayıda örnek kalacak şekilde bölünür.

Her bölmenin, ortalamayla ya da bölmenin en alt ve üst sınırlarıyla temsil edilmesi

Bölme Derinliği = 3

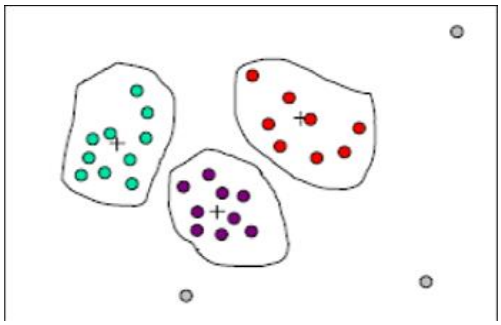
1. Bölme: 4, 8, 15
2. Bölme: 21, 21, 24
3. Bölme: 25, 28, 34

Ortalamayla düzeltme:
1. Bölme: 9, 9, 9
2. Bölme: 22, 22, 22
3. Bölme: 29, 29, 29

Alt-üst sınırla düzeltme:
1. Bölme: 4, 4, 15
2. Bölme: 21, 21, 24
3. Bölme: 25, 25, 34

- Kümeleme / Demetleme / Öbekleme (Clustering)

Benzer verilerin aynı kümede/öbekte olacak şekilde gruplanmasıdır.Bu kümelerin dışında kalan verilerin aykırılık olarak belirlenmesi ve silinmesidir.



- Eğri Uydurma (Curve Fitting)

Eğri uydurma, bir veri kümesine en iyi uyan matematiksel fonksiyonu bulma sürecidir. Özellikle gürültülü verileri düzelterek genel eğilimi belirlemek için kullanılır.

Bu yöntem, verilerin bir matematiksel modelle temsil edilmesini sağlar. Genellikle doğrusal (linear), polinom, üstel veya logaritmik fonksiyonlar kullanılır.

Tutarsız Veri Nasıl Düzeltilir ?

Yukarıda belirttiğimiz

> Bir yerde 'yaş=30-40 arası', başka yerde 'doğum tarihi = 01.02.1970'

Örneğini düşünelim. Burada tek bir tarih formatı ayarlayabiliriz, İstanbul şehrimiz İst olarak da kayıt edilmişse bunları İstanbul olarak değiştirip standardize edebiliriz, manuel olarak değiştirebiliriz, ortalama ile düzenleyebiliriz. Mantıksal bir hata olmaması için yaş gibi alanları günümüz yılı - doğum tarihi mantığı gibi doldurabiliriz.

```python
import pandas as pd

# Örnek veri
df = pd.DataFrame({
    'ID': [1, 2],
    'Ad': ['Ahmet', 'Ayşe'],
    'Doğum_Tarihi': ['1995-07-20', '20.07.1995'],
    'Şehir': ['İstanbul', 'İst']
})

# Tarih formatını düzeltme
df['Doğum_Tarihi'] = pd.to_datetime(df['Doğum_Tarihi'], dayfirst=True)

# Şehir isimlerini standardize etme
df['Şehir'] = df['Şehir'].replace({'İst': 'İstanbul'})

print(df)
```

Veri Düzeltme Normalizasyon

Normalizasyon, verilerin belirli bir ölçekte (genellikle 0-1 veya -1 ile 1 arasında) dönüştürülmesini sağlayan bir ön işleme yöntemidir. Amaç, farklı ölçeklerdeki verileri aynı ölçeğe getirmek ve analizleri daha doğru hale getirmektir.

Örn: Boy: 170 cm → 1.7 m Kilo: 80 kg 100m Koşu Süresi: 10.2 saniye

Bu değişkenlerin birbiriyle ölçek olarak uyumsuz olduğu görebiliriz metre, kg ve saniye olarak. Normalizasyon yaparak hepsini benzer bir ölçeğe getirebiliriz.

Neden Normalizasyon Yapılır?

- Farklı ölçeklerdeki değişkenleri kıyaslamak için
- Makine öğrenmesi modellerinin daha iyi çalışmasını sağlamak için
- Hesaplamalarda büyük değerlerin etkisini azaltmak için
- Aykırı değerlerin etkisini azaltmak için

- min-max normalizasyon

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A$$

- z-score normalizasyon

$$v' = \frac{v - mean_A}{stand\_dev_A}$$

- ondalık normalizasyon

$$v' = \frac{v}{10^j}$$ j: Max($|v'|$)<1 olacak şekildeki en küçük tam sayı

- Min-Max Normalizasyonu:

    Veriyi [0,1] arasına sıkıştırır.

Örneğin aşağıda min-max normalizasyon kullanarak boy ve kilo alanlarını normalize edelim.

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler

olimpiyat_veri = pd.DataFrame({
'Sporcu': ['Ali', 'Ayse', 'Mehmet', 'Zeynep', 'Hasan' ],
'Boy': [160, 175, 180, 155, 190], # cm
'Kilo': [60, 70, 80, 55, 90] # kg })

# Min-Max Normalizasyonu
scaler = MinMaxScaler()
olimpiyat_veri[['Boy', 'Kilo']] = scaler.fit_transform(olimpiyat_veri[['Boy', 'Kilo']])

print(olimpiyat_veri)
```

- Z-Score Normalizasyonu (Standartlaştırma):

    Veriyi ortalama 0, standart sapma 1 olacak şekilde dönüştürür.

- Ondalık Normalizasyonu (Decimal Scaling Normalization)

Ondalık normalizasyonu, verinin büyüklüğüne bağlı olarak her değeri 10'un kuvvetine bölerek ölçeklendirme işlemidir. Özellikle büyük sayıları daha küçük bir ölçeğe çekmek için kullanılır.Verinin 0 ile 1 arasında veya -1 ile 1 arasında olmasını sağlar.

Veri Görselleştirme

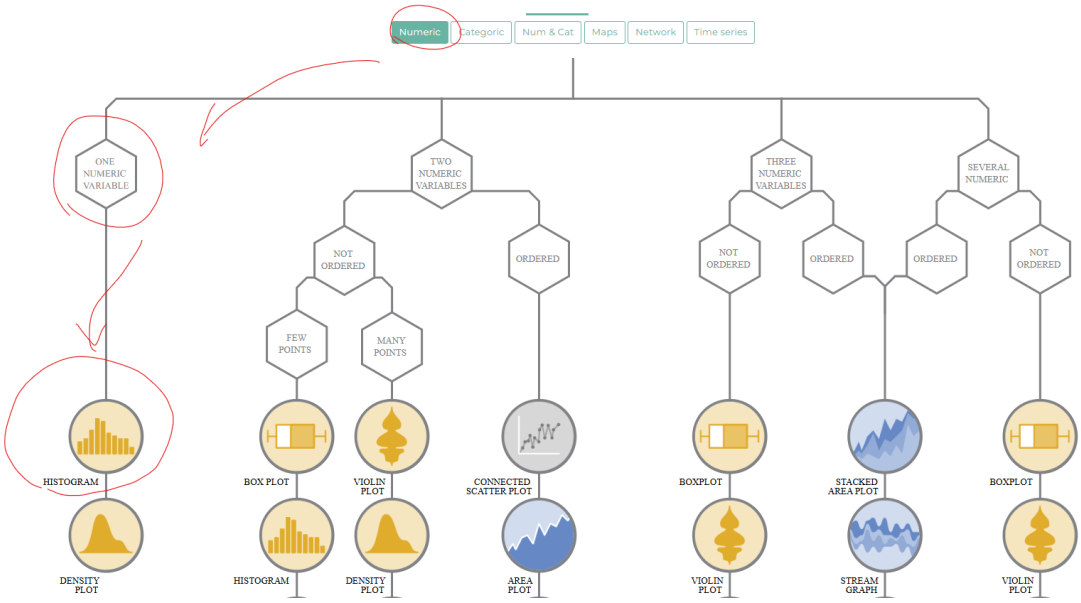
Elimizdeki veriler sayısal ve kategorik olarak ikiye ayrılmaktadır.

- Sayısal Değişkenler:
    1. Yas
    2. Boy
    3. Kilo
    4. Yil
- Kategorik Değişkenler:
    1. İsim
    2. Cinsiyet
    3. Takım
    4. UOK
    5. Sezon
    6. Sehir
    7. Spor
    8. Etkinlik
    9. Madalya

Sayısal verilerin veri sıklığı incelenecek. Teker teker değişkenlerin sıklığı ele alınacak.

data-to-viz.com, hangi durumda ne tarz grafik kullanılabilir olduğuna dair yol haritası sitesidir.

data-to-viz.com önerisine göre Sayısal > Tek Değişkenli değerler için HISTOGRAM grafiği uygulanacak.

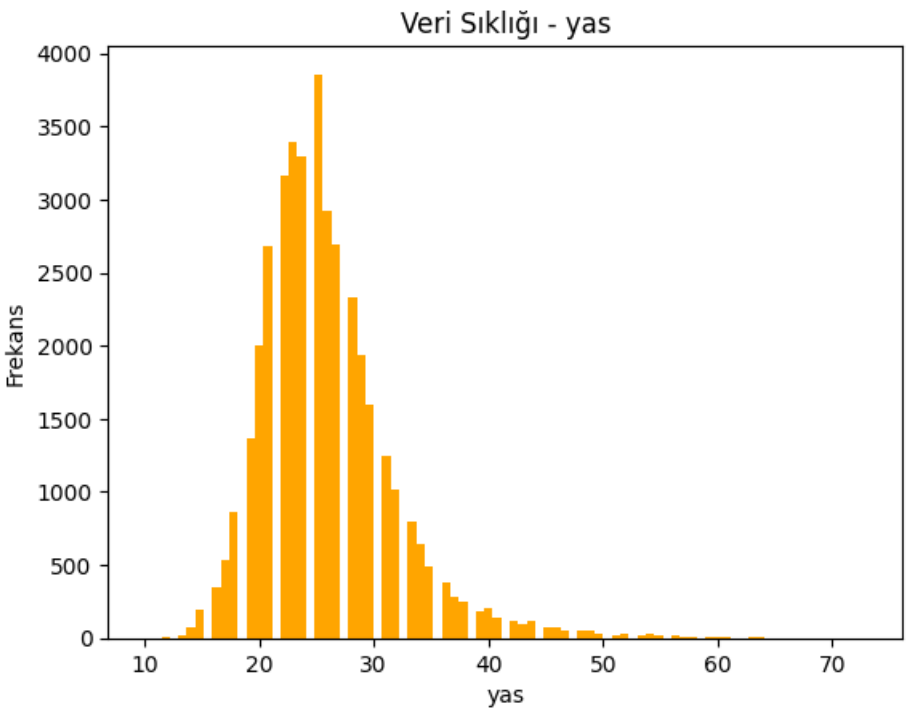


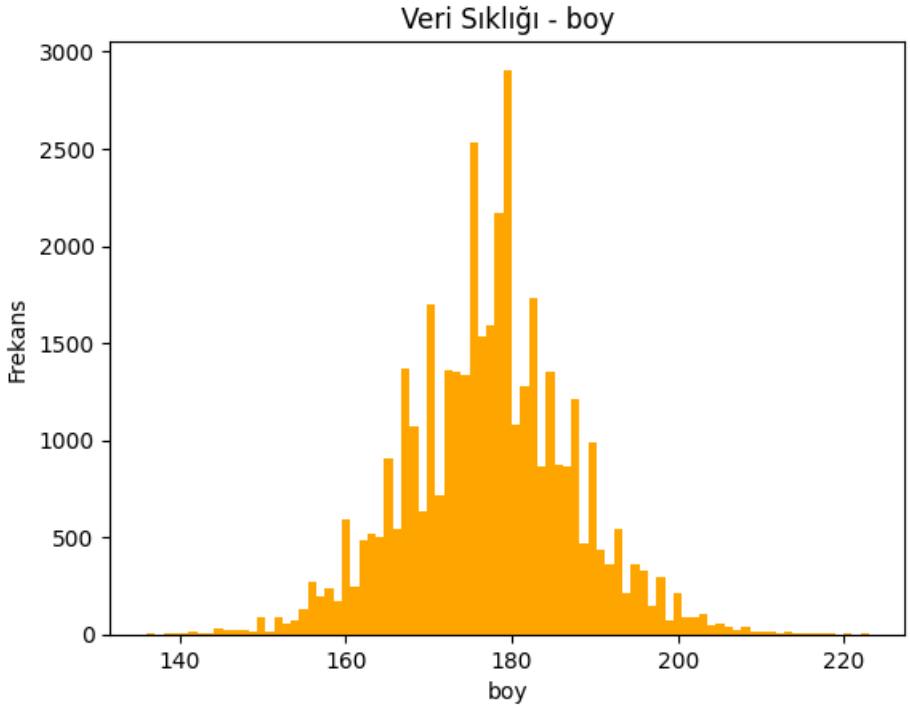
Tek Değişkenli Veri Analizi

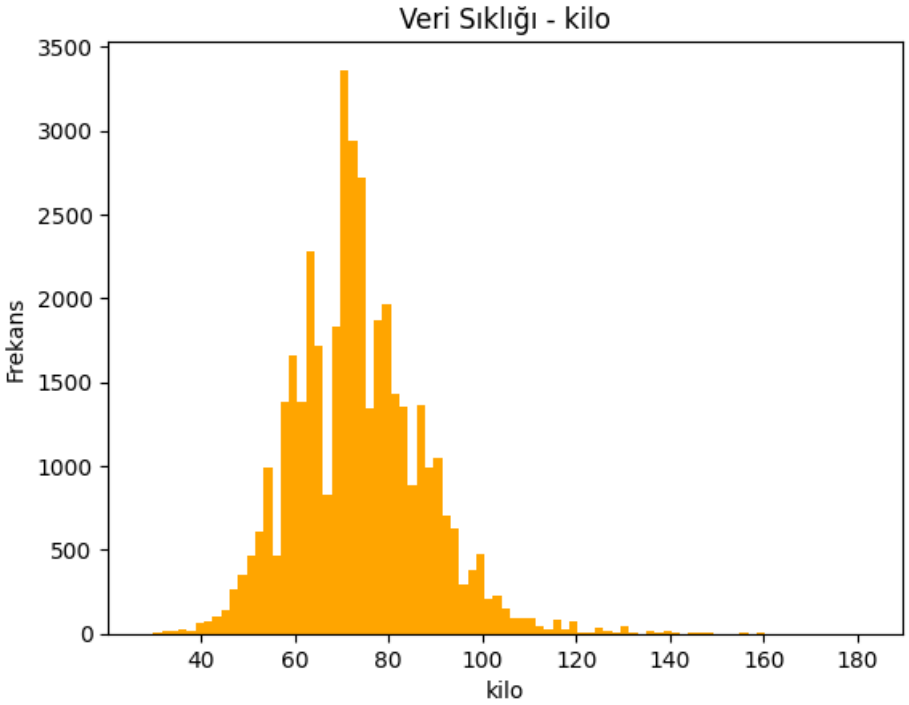
```
# Histogram grafiği çizdirecek fonksiyon
def plotHistogram(degisken):
    """
        Girdi: Değişken/Sütun Adı
        Çıktı: İlgili Değişkenin Histogramı
    """
    plt.figure()
    plt.hist(veri[degisken], bins=85, color="orange") # Histogram çizmek için plt.hist() kullanılmalı
    plt.xlabel(degisken)
    plt.ylabel("Frekans")
    plt.title("Veri Sıklığı - {}" .format(degisken))
    plt.show()
```

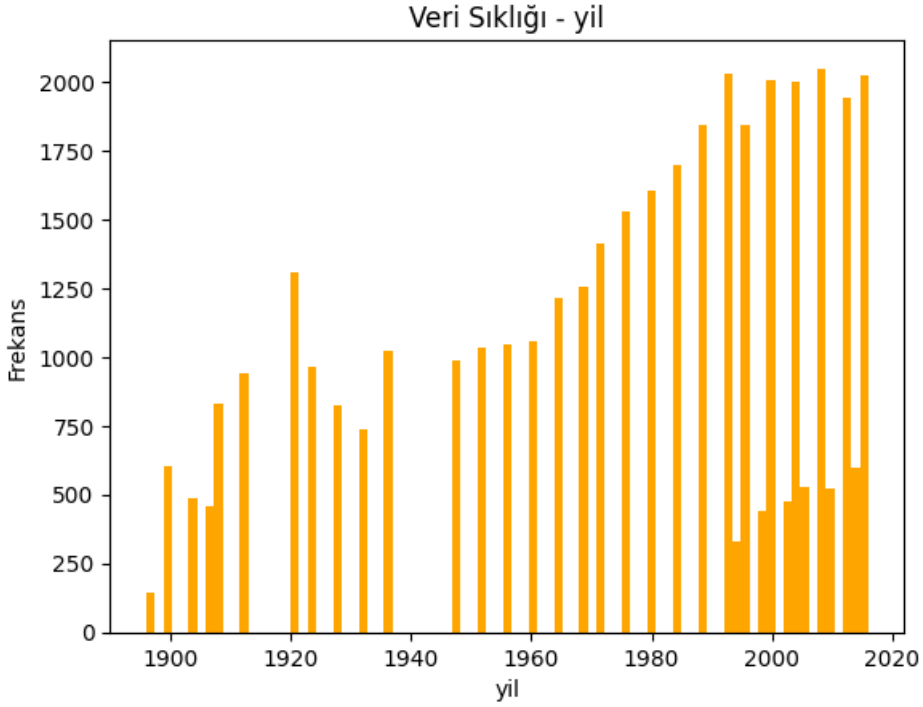
```
# tüm sayısal değişkenler için histogram çizdirelim
sayisal_degisken = ["yas", "boy", "kilo", "yil"]
for i in sayisal_degisken:
    plotHistogram(i)
```

- Yas, boy, kilo ve yil değişkenleri için histogram grafiği oluşturuldu.


Veri Sıklığı - yas
Frekans
yas
0
500
1000
1500
2000
2500
3000
3500
4000
10
20
30
40
50
60
70


Veri Sıklığı - boy
Frekans
boy
0
500
1000
1500
2000
2500
3000
140
160
180
200
220


Veri Sıklığı - kilo
Frekans
kilo
0
500
1000
1500
2000
2500
3000
3500
40
60
80
100
120
140
160
180


Veri Sıklığı - yil
Frekans
yil
0
250
500
750
1000
1250
1500
1750
2000
1900
1920
1940
1960
1980
2000
2020

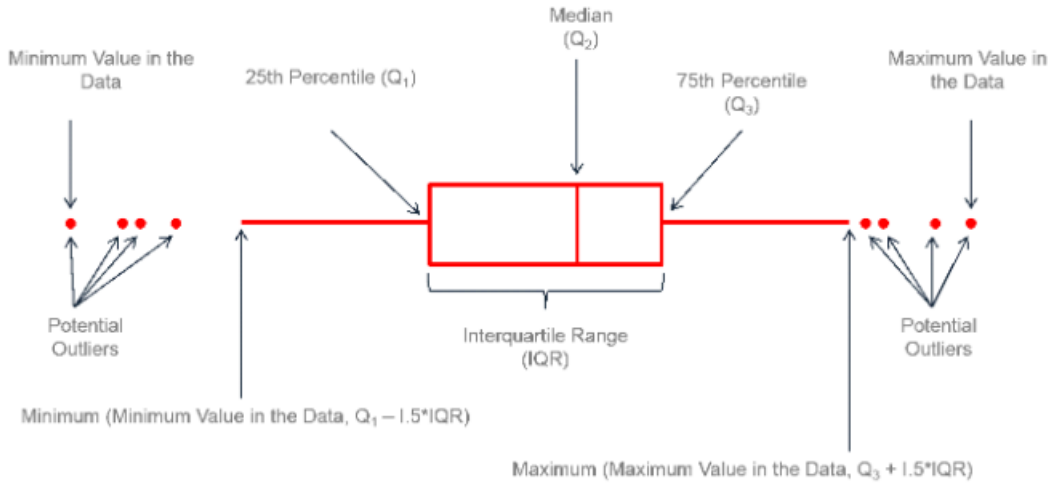
Diğer Veri Görselleştirme Grafikleri ve Özellikleri

1. Histogram:

a. Neden Kullanılır?
   i. Sürekli verilerin dağılımını göstermek için kullanılır.
   ii. Veri setindeki frekansları analiz etmek için idealdir.

b. Nasıl Kullanılır?
   i. Veriler belirli aralıklara (bin) bölünerek her aralığın frekansı çubuklarla gösterilir.
   ii. Örneğin, müşteri yaş dağılımını görmek isteyen bir firma, histogram kullanarak en çok hangi yaş gruplarında müşterileri olduğunu analiz edebilir.

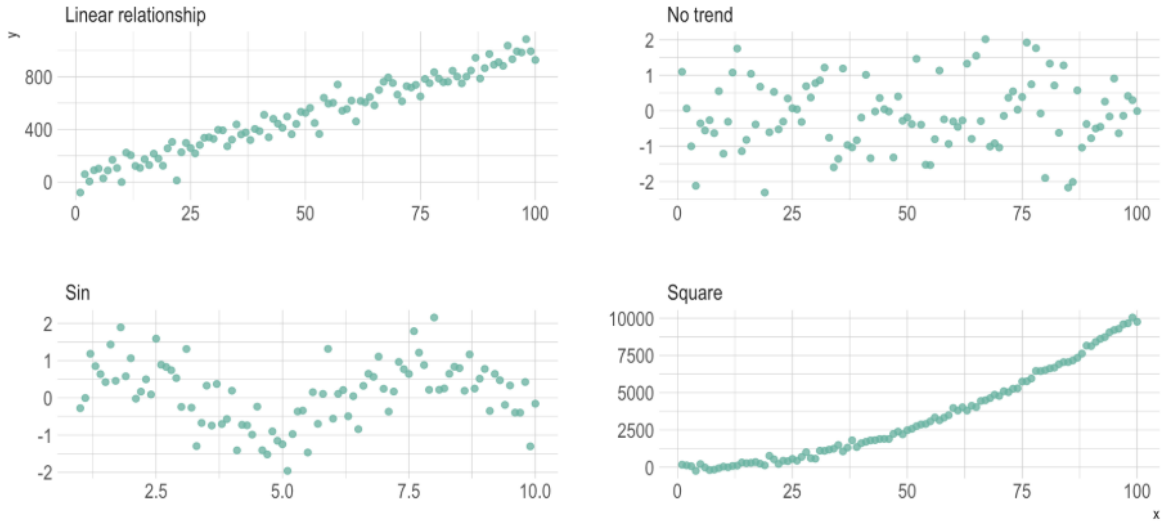
2. IQR (Çeyrekler Arası Aralık) ve Box Plot (Kutu Grafiği)



- Boxplot, verinin nasıl dağıldığını görmek için kullanılan bir grafik türüdür.
- Özellikle ortalama, yayılım ve aykırı değerleri gösterir.
- **Boxplot'un Bölümleri**

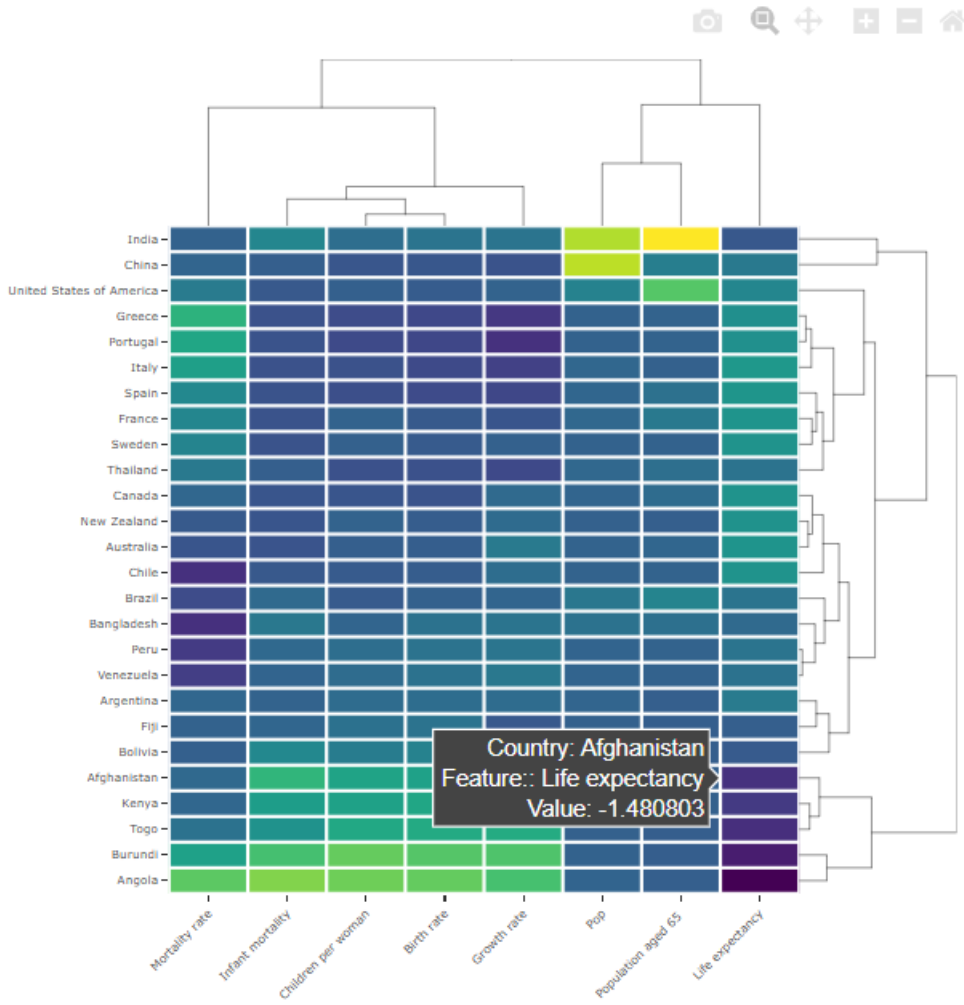
1. **Medyan Çizgisi (Ortanca Değer):**
   - Kutunun içindeki çizgidir.
   - Veriyi ikiye böler: Yarısı bu çizginin altında, yarısı üstündedir.
2. **Kutunun Uçları (Q1 & Q3 - Çeyrekler):**
   - **Alt uç (Q1)** → Verinin %25'inin altına düştüğü noktadır.
   - **Üst uç (Q3)** → Verinin %75'inin altına düştüğü noktadır.
   - Yani, kutu verinin ortadaki %50'sini gösterir.
3. **IQR (Çeyrekler Arası Mesafe):**
   - **Q3 - Q1** farkıdır.
   - Verinin ne kadar yayılmış olduğunu anlamamızı sağlar.
4. **Bıyıklar (Whiskers):**
   - Verinin minimum ve maksimum değerlerini gösterir.
   - Ama sadece **çok uç noktalara** kadar gider, daha uç noktalar "aykırı" kabul edilir.
5. **Aykırı Değerler (Outliers):**
   - Bıyıkların dışında kalan noktalardır.
   - Veri setinde **anormal ya da sıradışı** değerleri gösterir.

3. Scatter Plot (Saçılım Grafiği)



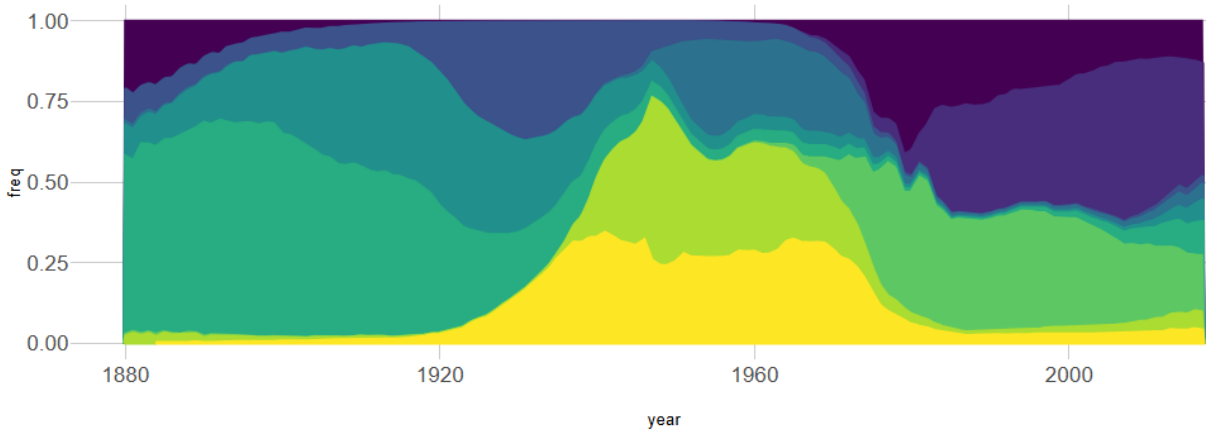
- Scatter plot, iki sayısal değişken arasındaki ilişkiyi gösteren bir grafik türüdür.
- X ekseni ilk degiskeni, Y ekseni ikinci degiskeni temsil eder.
- Korelasyon katsayısı ile değişkenler arasındaki bağlantı incelenir.
- Grafikten doğrusal (linear) veya farklı türde ilişkiler tespit edilebilir.
- Makine öğrenimi modellerinde bağımlı değişken (Y) ile bağımsız değişken (X) arasındaki ilişkiyi anlamak için sıkça kullanılır.

4. Heatmap (Isı Haritası)



- Heatmap, matris formatında gösterilen verilerin, renk yoğunluğu ile ifade edilerek görselleştirilmesini sağlayan bir grafik türüdür.
- Yukarıdaki örnekte; satırlar ülkeleri, sütunlar ise demografik özellikleri temsil eder.
- Dendrogram (ağaç yapısı), benzer özelliklere sahip ülkeleri gruplamak için kullanılmıştır.
- Örneğin, India ve China aynı kümede bulunuyor. Muhtelemen büyük nüfusları yüzünden aynı değere sahipler.
- Örneğin, Afghanistan'ın Life Expectancy değeri -1.48 olarak belirlenmiş yani bu metrik muhtelemen normalleştirilmiştir.

5. Stacked Area Graph (Yığılmış Alan Grafiği)



Stacked area graph, birden fazla grubun zaman içindeki değişimini aynı grafikte gösteren bir alan grafiği türüdür.

Her grubun değeri üst üste eklenerek gösterildiği için:

- Toplam değişimi izlemek mümkündür.
- Her bir grubun zaman içindeki değişimi ve katkısı görülebilir.

Veri Görselleştirmede Renklerin Kullanımı

Doğru renk paleti çok önemlidir.

Bunun için Konstrast Renkler Tablosuna bakılarak birbirleriyle 0.40 üzeri birlikte kullanılabilir renkler kullanılmalıdır.

	Siyah	Beyaz	Kırmızı	Yeşil	Mavi	Çivit	Eflatun	Turuncu	Sarı
Siyah	0.00	1.00	0.30	0.59	0.11	0.70	0.41	0.60	0.89
Beyaz	1.00	0.00	0.70	0.41	0.89	0.30	0.59	0.41	0.11
Kırmızı	0.30	0.70	0.00	0.29	0.19	0.40	0.11	0.30	0.59
Yeşil	0.59	0.41	0.29	0.00	0.48	0.11	0.18	0.01	0.30
Mavi	0.11	0.89	0.19	0.48	0.00	0.59	0.30	0.49	0.78
Çivit	0.70	0.30	0.40	0.11	0.59	0.00	0.29	0.11	0.19
Eflatun	0.41	0.59	0.11	0.18	0.30	0.29	0.00	0.19	0.48
Turuncu	0.60	0.41	0.30	0.01	0.49	0.11	0.19	0.00	0.30
Sarı	0.89	0.11	0.59	0.30	0.78	0.19	0.48	0.30	0.00

Kullanılan Kaynaklar:

https://www.data-to-viz.com/

https://www.btkakademi.gov.tr/portal/course/saglikta-yapay-zeka-34994

https://www.btkakademi.gov.tr/portal/course/veri-bilimi-ve-makine-ogrenmesi-atolyesi-bootcamp-2022-19100

https://www.btkakademi.gov.tr/portal/course/uygulamali-kaggle-27841