MAKİNE ÖĞRENMESİ

Ödev ve Proje Dosyası

Emir CAN

21410082015

2023-2024 Eğitim Öğretim Yılı

|  |  |
| --- | --- |
| Öğrenci No: | 21410082015 |
| İsim soyisim: | Emir CAN |
| Ödev numarası: | 131023K-1 |
| Tarih: | 13 Ekim 2023 |
| Ödev sorusu : | Klasörde verilen m1 ve m2 txt dosyalarının içindeki metinlerin benzeşim oranını buldurunuz? |
| str1 = open('m1.txt','r').read()  str2 = open('m2.txt','r').read()  str1 = str1.replace(".","")  str1 = str1.replace(",","")  str1 = str1.replace("'","")  str2 = str2.replace(".","")  str2 = str2.replace(",","")  str2 = str2.replace("'","")  durma = ['aha','ahacık','ama','başka','belli','beri','bilfarz','değin','dek','denli','derece','doğru','evet','gibi','ha','hayır','he','için','ila','işte','kadar','kelli',  'mesela','oldu','olur','örneğin','peki','rağmen','söz gelimi','söz gelişi','söz temsili','tamam','temsil','üzere','üzre','ve','veya','ya']  for i in durma:  str1=str1.replace(i,"")  str2=str2.replace(i,"")  l1 = list(str1.split(" "))  print(l1)  print(len(l1))  l2 = list(str2.split(" "))  print(l2)  print(len((l2)))  s1 = set(l1)  print(s1)  print(len(s1))  s2 = set(l2)  print(s2)  print(len(s2))  st = set.union(s1,s2)  print(st)  print(len((st)))  ts = len(s1)+len(s2)  print("iki metin için tekil sözcük sayısı : ", ts)  tss = len(st)  print("Birleşim sonrası tekil sözcük sayısı : " , tss)  fark = ts-tss  print("fark : ", fark)  benzeme = fark\*100/tss  print("Benzeme Oranı : %" ,benzeme) | |

|  |  |
| --- | --- |
| Öğrenci No: | 21410082015 |
| İsim soyisim: | Emir CAN |
| Ödev numarası: |  |
| Tarih: | 19 Ekim 2023 |
| Ödev sorusu : | Araştırma ödevi: Naive Bayes yöntemi hakkında araştırma yaparak, küçük bir özet anlatım yapınız. Kod ödevi girilen metnin Shannon Entropy değerini hesaplayarak bit şeklinde dönüştürünüz. |
| Naive Bayes : Naive Bayes sınıflandırıcısının temeli Bayes teoremine dayanır. bir öğrenme algoritmasıdır aynı zamanda dengesiz veri kümelerinde de çalışabilir. Algoritmanın çalışma şekli bir eleman için her durumun olasılığını hesaplar ve olasılık değeri en yüksek olana göre sınıflandırır. Az bir eğitim verisiyle çok başarılı işler çıkartabilir.  import math  s=input("Ad soyad Giriniz : ")  s=s.replace(" ","")  liste = []  for i in s:  liste.append(i)  print(liste)  k = set(liste)  print(k)  d={}  for i in k:  adet = liste.count(i) # girilen harften kaç tane var  oran = adet/len(liste)  d.update({i:oran})  print(d)  shannon = 0  for i in d:  v = d[i]  shannon += v\*math.log2(v) #Shannon Entropy formulü  shannon\*=-1  print(shannon)  bs = math.ceil(shannon) # bit sayısını yukarı yuvarlıyoruz  print("Bit Sayısı : ",bs)  dk = list(k)  b=[]  for i in range(int(math.pow(2,bs))): # 2'nin üssü bs kadar döngü  a=bin(i)[2:] #10'luk tabanda verilen sayıyı 2'lik tabana çevirir  b.append(a)  dk.sort() #sıralar  print(dk)  print(b)  for i in range (len(b)):  for j in range (bs - len(b[i])):  b[i]="0"+b[i] # kaç tane eksik 0 varsa ekler  print(b)  coded = ""  for i in liste:  coded += b[dk.index(i)]  print(coded) | |

|  |  |
| --- | --- |
| Öğrenci No: | 21410082015 |
| İsim soyisim: | Emir CAN |
| Ödev numarası: |  |
| Tarih: | 26 Eki 2023 |
| Ödev sorusu : | Bulduğunuz basit bir veri seti üzerinde Gaussian Naive Bayes üzerinde prediction yaptırınız. |
| C:\Users\can\Desktop\veri madenciliği\Örnek+Veri+Seti.jpg  C:\Users\can\AppData\Local\Packages\Microsoft.Windows.Photos_8wekyb3d8bbwe\TempState\ShareServiceTempFolder\indir.jpeg  P(Evet)=9/14  P(Hayır)=5/14  P(<30,Evet)=2/9  P(<30,Hayır)=3/5  P(Yüksek,Evet)=2/9  P(Yüksek,Hayır)=2/5  P(Hayır,Evet)=3/9  P(Hayır,Hayır)=4/5  P(Vasat,Evet)=6/9  P(Vasat,Hayır)=2/5  Oranları çarpalım(Evet)  P(X|Bilgisayar alır=Evet)=2/9*2/9*3/9*6/9*9/14=0,0070  Oranları çarpalım(Hayır)  P(X|Bilgisayar alır=Hayır)=3/5*2/5*4/5*2/5*5/14=0,0274  P(X|Bilgisayar alır=Evet)=0,0070/(0,0070+0,0274)=0,2034  P(X|Bilgisayar alır=Hayır)=0,0274/(0,0274+0,0070)=0,7965  import numpy as np  from sklearn.naive\_bayes import GaussianNB  #sklearn den Naive Bayes'i import ediyoruz  X = np.array([                [0, 0, 0, 0],   #Tablodaki verileri giriyoruz                [0, 0, 0, 1],                [1, 0, 0, 0],                [2, 1, 0, 0],                [2, 2, 1, 0],                [2, 2, 1, 1],                [1, 2, 1, 1],                [0, 1, 0, 0],                [0, 2, 1, 0],                [2, 1, 1, 0],                [0, 1, 1, 1],                [1, 1, 0, 1],                [1, 0, 1, 0],                [2, 1, 0, 1]  ])  Y = np.array([0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0])  #Tablodaki çıktıları giriyoruz  model = GaussianNB()    #Gauss Naive Bayes fonksiyonunu modele tanımladık  model.fit(X, Y)         #Girdi ve çıktıları atadık  GaussianNB()  print(model.predict([[1, 2, 0, 1]])) | |

|  |  |
| --- | --- |
| Öğrenci No: | 21410082015 |
| İsim soyisim: | Emir CAN |
| Ödev numarası: |  |
| Tarih: | 26 Eki 2023 |
| Ödev sorusu : | 1-k- En Yakın Komşuluk Algoritması (k- Nearest Neighbors Algorithm / k-NNA) yöntemi hakkında ön okuma yapın  2-Bulduğunuz basit bir veri seti üzerinde KNN üzerinde prediction yaptırınız. |
| K-NN : K en yakın komşu (KNN) algoritması, adından da anlaşılacağı üzere, komşularına bakarak tahminlemede bulunan bir algoritmadır. KNN algoritmasında, benzer olan şeyler birbirine yakındır varsayımı geçerlidir. Bir KNN modeli, bir grafik üzerindeki iki nokta arasındaki mesafeyi kullanarak benzerliği hesaplar. Noktalar arasındaki mesafe ne kadar büyükse, o kadar az benzerdirler. Noktalar arasındaki mesafeyi hesaplamanın birçok yolu vardır, ancak en yaygın mesafe ölçüsü sadece Öklid mesafesidir  import math  x1=[5.5,2.2,2.4,4.7,3.8,6.3,7.8,6.6,1.4,2.9]  x2=[2.1,3.1,8.5,7.3,6.9,6.7,5.5,9.6,6.8,4.2]  y=["A","B","B","A","B","A","A","A","B","B"]  a = float(input("Lütfen x1 değerini giriniz : "))  b = float(input("Lütfen x2 değerini giriniz : "))  k = int(input("Lütfen k değerini giriniz : "))  di = {}  for i in range(len(x1)):  afark = x1[i]-a  bfark = x2[i]-b  u = math.sqrt(math.pow(afark,2)+math.pow(bfark,2))  di.update({i:u})  print(di)  dis = {k:v for k, v in sorted(di.items(), key=lambda item: item[1]) }  print(dis) #En küçük değer ne ise onun key değerini al ve yeni dis'e at  for i in dis:  print(x1[i]," ",x2[i]," ",i," ",y[i]," ",dis[i])  dis\_list = list(dis)  k\_list = []  for i in range(k):  k\_list.append(y[dis\_list[i]])  print(k\_list)  A=B=0 #Sayaç  A=k\_list.count("A")  B=k\_list.count("B")  if A>B :  print("Sınıflandırma sonucu = A")  else:  print("Sınıflandırma sonucu = B") | |

|  |  |
| --- | --- |
| Öğrenci No: | 21410082015 |
| İsim soyisim: | Emir CAN |
| Ödev numarası: |  |
| Tarih: | 9 Kasım 2023 |
| Ödev sorusu : | Yapay sinir ağlarını kullanarak kendi verinizi eğitin |
| a1=[0,0,1,2,2,2,1,0,0,2,0,1,1,2]  a2=[0,0,0,1,2,2,2,1,2,1,1,1,0,1]  a3=[0,0,0,0,1,1,1,0,1,1,1,0,1,0]  a4=[0,1,0,0,0,1,1,0,0,0,1,1,0,1]  cikti=[[0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0]]  agirlik = [1.20,1.80,1.40,1.50]  fi = 0  lmbda = 0.3  NET = 0  output = 0  def net\_hesapla(i):      NET = a1[i]\*agirlik[0]+a2[i]\*agirlik[1]+a3[i]\*agirlik[2]+a4[i]\*agirlik[3];      return NET  def perceptron\_cikti(net):      if net>fi:         output=1      else:         output=0      return output  def agirliklari\_yeniden\_hesapla(o,i):      if o != cikti[i]:          agirlik[0] = agirlik[0] - lmbda \* a1[i]          agirlik[1] = agirlik[1] - lmbda \* a2[i]          agirlik[2] = agirlik[2] - lmbda \* a3[i]          agirlik[3] = agirlik[3] - lmbda \* a4[i]  for i in range(1):      for j in range(len(cikti)):          a = net\_hesapla(j);          op = perceptron\_cikti(a)          agirliklari\_yeniden\_hesapla(op,j)  print("Eğitim tamamlandı!\nEğitim sonrası ağırlıklar:")  print(agirlik)  g1, g2, g3, g4 = input("Lütfen test değerlerini giriniz:").split()  a1[0]=int(g1)  a2[0]=int(g2)  a3[0]=int(g3)  a4[0]=int(g4)  print(perceptron\_cikti(net\_hesapla(0))) | |

|  |  |
| --- | --- |
| Öğrenci No: | 21410082015 |
| İsim soyisim: | Emir CAN |
| Ödev numarası: |  |
| Tarih: | 16 Kas 2023 |
| Ödev sorusu : | Karar ağaçları kullanarak kendi verinizi eğitin |
| import pandas as pd  import numpy as np  import seaborn as sns  import matplotlib.pyplot as plt  from google.colab import files  uploaded = files.upload()  import pandas as pd  import io  data = pd.read\_excel('penguins\_veri.xlsx')  print(data.head())    df = pd.DataFrame(data)  data.sex=[1 if x=="MALE" else 0 for x in data.sex]  data.island = [0 if x == "Torgersen" else 1 if x == "Biscoe" else 2 for x in data.island]  data.species = [0 if x == "Adelie" else 1 if x == "Gentoo" else 2 for x in data.species]  data    # Hedef değişken ve bağımsız değişkenleri belirleme  X = df[['island', 'culmen\_length\_mm', 'flipper\_length\_mm', 'body\_mass\_g', 'sex']]  y = df['species']  # Veriyi eğitim ve test setlerine bölelim  from sklearn.model\_selection import train\_test\_split  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2)  # Karar ağacı modelini oluşturma  from sklearn.tree import DecisionTreeRegressor  model = DecisionTreeRegressor(random\_state=42)  model.fit(X\_train, y\_train)  # Test seti üzerinde tahmin yapma  y\_pred = model.predict(X\_test)  # Model performansını değerlendirme  print(model.score(X\_train,y\_train))  print(model.score(X\_test,y\_test))  1.0  0.9028985507246376  new\_data = {#328'de ki veriler      'island': [1],      'culmen\_length\_mm': [47.2],      'flipper\_length\_mm': [214],      'body\_mass\_g': [4925],      'sex': [0]}  new\_df = pd.DataFrame(new\_data)  # Tahmin yapın  predicted\_price = model.predict(new\_df)  print(f"Penguen türü tahmini: {predicted\_price[0]}")  Penguen türü tahmini: 1.0 | |
| Öğrenci No: | 21410082015 |
| İsim soyisim: | Emir CAN |
| Ödev numarası: |  |
| Tarih: | 14 Ara 2023 |
| Ödev sorusu : | Destek Vektör Makineleri kullanarak kendi veri setinizi eğitin |
| import numpy as np  import matplotlib.pyplot as plt  from sklearn import svm  x = np.array([ [3, 2],[2, 3],          [3, 3], [4, 2],          [4, 5], [5, 4],          [4, 4], [5, 3]])  # çıktı değişkeninin 8 değeri bulunmakta  y = np.array([-1, -1, -1, -1, 1, 1, 1, 1])  # göz ile bir plot çizelim  # çıktı değişkenleri girdi üzerinde gösteren plot  # eğer -1'e eşitse girdinin gösterimi "-" ile  # eğer 1'e eşitse girdinin gösterimi "+" olarak gösterilir.  for index, deger in enumerate(x):      if y[index] == -1:          plt.scatter(deger[0], deger[1]                      ,s=100, marker='\_'                      ,linewidths=5)      else:          plt.scatter(deger[0], deger[1]                      ,s=100, marker='+'                      ,linewidths=5)  # varsayımsal SVM plot  plt.plot([1, 6],           [6, 1])  plt.show() | |
| Öğrenci No: | 21410082015 |
| İsim soyisim: | Emir CAN |
| Ödev numarası: |  |
| Tarih: | 14 Ara 2023 |
| Ödev sorusu : | K-Means kullanarak kendi verinizi eğitin |
| from sklearn.cluster import KMeans  import numpy as np  import matplotlib.pyplot as plt  C:\Users\can\AppData\Local\Packages\Microsoft.Windows.Photos_8wekyb3d8bbwe\TempState\ShareServiceTempFolder\Ekran Alıntısı.jpeg  X=np.array([[1.0,1.5],[1.0,4.5],[2.0,1.5],[2.0,3.5],[3.0,2.5],[5.0,6.0]])  kmeans=KMeans(n\_clusters=2,random\_state=0).fit(X)  kmeans.labels\_  kmeans.predict([[4.0,2.5],[3.0,1.5]])  kmeans.cluster\_centers\_  fig,ax=plt.subplots()  ax.plot(X,'bo')  plt.show() | |

T.C

NECMETTİN ERBAKAN ÜNİVERSİTESİ

UYGULAMALI BİLİMLER FAKÜLTESİ

YÖNETİM BİLİŞİM SİSTEMLERİ BÖLÜMÜ

İKİNCİ ÖĞRETİM

MAKİNE ÖĞRENMESİ

FİNAL ÖDEVİ

ARAŞTIRMA KONUSU

Karar Ağaçları ile Öğrencinin Becerilerine Göre Meslek Önerme

ÖDEV SAHİBİ

Emir CAN

21410082015

1. MAKİNE ÖĞRENMESİ NEDİR?

Makine öğrenimi, bilgisayar sistemlerinin deneyimlerini veriyle geliştirmek ve örüntüleri belirlemek için kullanılan yapay zeka alt dalıdır. Bu sistemler, verilerden öğrenerek belirli bir görevi veya problemi gerçekleştirebilen algoritmaları içerir.

Makine öğrenimi süreci şu adımları içerir:

1. **Veri Toplama:** İşlenecek veri kümesi toplanır. Bu veriler, amaca yönelik olarak etiketlenmiş (denetimli öğrenme), etiketsiz (denetimsiz öğrenme) veya karışık veri tipleri olabilir.
2. **Veri Ön İşleme:** Toplanan veriler temizlenir, eksik değerler doldurulur, gereksiz özellikler kaldırılır ve veri ölçeklendirme veya dönüşümler gibi işlemler uygulanır.
3. **Model Seçimi:** Veri kümesine uygun bir model seçilir. Sınıflandırma, regresyon, kümeleme veya boyut azaltma gibi çeşitli makine öğrenimi algoritmaları bulunmaktadır. Bu algoritmalardan uygun olanı seçilir.
4. **Model Eğitimi:** Seçilen model, veri kümesi üzerinde eğitilir. Bu, modelin girdi ve çıktılar arasındaki ilişkileri öğrenmesini sağlar. Eğitim sürecinde model, veri üzerinde iteratif olarak çalışarak kendini geliştirir.
5. **Model Değerlendirme:** Eğitilmiş model, ayrı bir test veri seti üzerinde değerlendirilir. Modelin ne kadar iyi performans gösterdiği, doğruluk, hassasiyet veya başka ölçütlerle değerlendirilir.
6. **Model İyileştirme:** Modelin performansını artırmak için gerektiğinde hiperparametreler ayarlanır veya farklı özellikler eklenir.

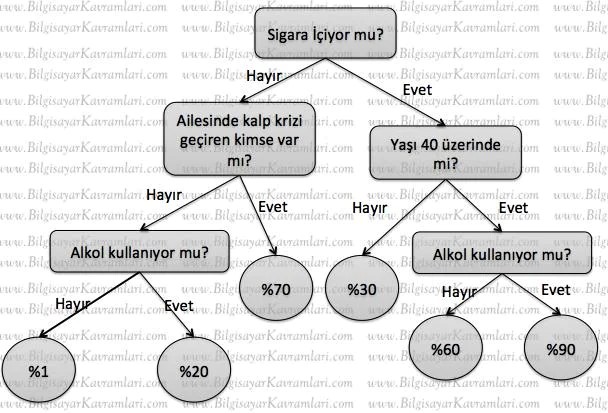
2.KARAR AĞAÇLARI NEDİR

Karar ağaçları, özellikle sınıflandırma ve regresyon görevlerinde kullanılan bir makine öğrenimi algoritmasıdır. Karar ağaçları, veri setini bir dizi karar kuralı ve sonuçlandırma düğümleri ile temsil eden bir ağaç yapısı oluşturur. Bu ağaç yapısı, veri setini belirli koşullar altında segmentlere böler ve her segment için bir tahmin yapar.

Karar ağaçları, veri setini bölerek bilgi kazancını maksimize etmeye çalışan bir algoritma kullanır. Temel amaç, her bölümün homojen bir şekilde sınıflandırılmasını veya regresyonunu sağlamaktır. Bilgi kazancı, veri setinin belirli bir özelliği tarafından bölünmesinin ardından entropy (belirsizlik ölçüsü) veya Gini impurity (Gini belirsizliği) gibi ölçümlerle ölçülür.

Karar ağacının temel bileşenleri şunlardır:

1. **Kök Düğüm (Root Node):** Ağacın başlangıç düğümüdür ve veri setini bölerek diğer düğümlere yönlendiren ilk kararı içerir.
2. **Bölme Düğümleri (Split Nodes):** Kök düğümden sonra gelen ve veri setini bölerek alt düğümlere yönlendiren düğümlerdir. Her bir bölme düğümü, belirli bir özelliğin değerine göre veri setini iki veya daha fazla parçaya böler.
3. **Yaprak Düğümleri (Leaf Nodes):** Karar ağacının son düğümleridir ve her biri bir sınıf veya regresyon tahmini içerir. Yaprak düğümleri, bir veri setinin sonlandığı ve bir tahminin yapıldığı yerlerdir.



### Avantajlar:

1. **Anlaşılabilirlik:**
   * Karar ağaçları, insanlar tarafından anlaşılması kolay bir yapıya sahiptir. Ağaç yapısı, kararları adım adım gösterdiği için yorumlanması kolaydır.
2. **Değişken Önemi:**
   * Her bir özelliğin modeldeki önem düzeyi kolayca belirlenebilir. Bu, hangi özelliklerin modelin tahminlerinde daha etkili olduğunu anlamak açısından değerlidir.
3. **Çeşitli Görevlerde Kullanılabilirlik:**
   * Sınıflandırma ve regresyon gibi birçok görevde kullanılabilirler.
4. **Parametre Ayarları:**
   * Modelin karmaşıklığını kontrol etmek için çeşitli parametre ayarlarına sahiptirler. Bu, aşırı uydurmayı kontrol etmeye yardımcı olabilir.
5. **Kategorik ve Sayısal Verilere Uyum:**
   * Hem kategorik hem de sayısal veri tiplerini işleyebilirler.

### Dezavantajlar:

1. **Overfitting Eğilimi:**
   * Eğitim verilerine çok sıkı uyarlanabilirler (overfitting). Bu durumu kontrol etmek için ağacın derinliği veya minimum örnekleme sayısı gibi parametreler ayarlanmalıdır.
2. **Duyarlılık:**
   * Giriş verilerindeki küçük değişikliklere duyarlı olabilirler, bu nedenle giriş verilerindeki hafif değişiklikler model sonuçlarını önemli ölçüde etkileyebilir.
3. **Bağlılık:**
   * Karar ağaçları, ağaç üzerindeki bir düğümdeki bir değişikliğin genel model performansını etkileyebilir, bu nedenle tek bir ağacın genel performansını artırmak için birden çok ağacın bir araya getirildiği bir "enseble" yöntemi olan RandomForest gibi yöntemler daha sık tercih edilebilir.
4. **Veri Dengesizliği:**
   * Veri setinde dengesiz sınıflar varsa, ağaçlar çoğunluk sınıfına odaklanabilir ve azınlık sınıfındaki örnekleri yanlış sınıflandırabilir.

## 3.Veri Kümesi Hakkında

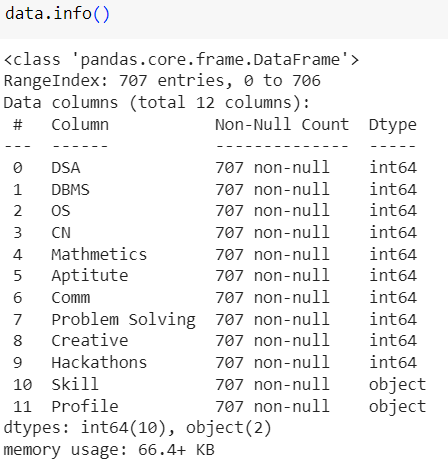
Bu veri seti, öğrencilerin teknik ve sosyal becerilerinin yanı sıra hackathonlara katılımları ve belirli araçlar veya dillerdeki yeterlilikleri hakkında ayrıntılı bir görünüm sunar. Öğrenciler arasındaki beceri eğilimlerini ve işe yerleştirme uygunluğunu analiz etmekle ilgilenen eğitim kurumları, işe alım görevlileri veya araştırmacılar için idealdir. Veri seti; Veri Yapıları ve Algoritmalar , Veritabanı Yönetim Sistemleri, İşletim Sistemleri, Matematik ve çeşitli sosyal beceriler gibi alanları kapsayarak her öğrencinin yeteneklerine ilişkin bütünsel bir görünüm sağlar.

Bu veri seti, öğrencilerin becerilerini ve ilgi alanlarını analiz etmek için kullanılabilir ve öğrencilerin UI/UX, yazılım mühendisi, veri bilimi vb. gibi farklı profesyonel profillere uygunluklarının belirlenmesine yardımcı olabilir.

4.VERİ SETİ İÇERİĞİ

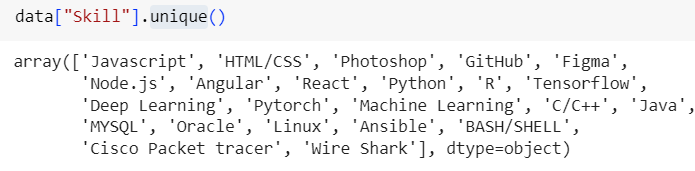


* DSA: Veri Yapıları Algoritmalar
* DBMS: Veritabanı Yönetim Sistemleri
* OS: İşletim Sistemleri
* CN: Bilgisayar Ağları
* Mathmetics: Matematik Becerileri
* Aptitute: Genel Yetenek
* Comm: İletişim Yetenekleri
* Problem Solving: Problem çözme yetenekleri
* Creative: Yaratıcılık düzeyi
* Hackathons: Hackathonlara katılım düzeyi
* Skill: Belirli araçlar veya dillerdeki yeterlilikler
* Profile: Potansiyel profesyonel profil uygunluğu

  
Veri setimiz 707 satır ve 12 sütundan oluşmaktadır. Veri setinde eksik değerler yoktur.

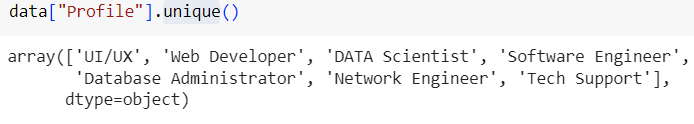
2 tane object türünde verimiz var.

Skill sütununda olan değerler:



Skill sütunundaki verileri sayısal değere dönüştürdüğümüzde aldıkları değerler:

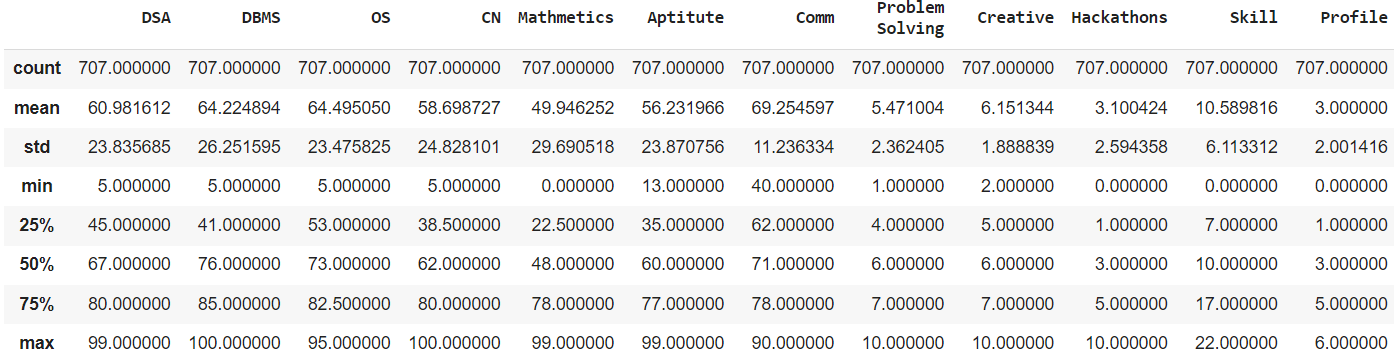
 0=Angular, 1=Ansible, 2=BASH/SHELL, 3=C/C++,  4=Cisco Packet tracer, 5=Deep Learning, 6=figma, 7=Github, 8=HTML/CSS,  9=Java, 10=javascprit, 11=Linux, 12=Mysql, 13=Machine Learning, 14=Node.js, 15=Oracle, 16=photoshop, 17=Python, 18=Pytorch, 19=R, 20=react, 21=Tensorflow, 22=Wire Shark

Profile sütununda olan değerler: 

Profile sütunundaki verileri sayısal değere dönüştürdüğümüzde aldıkları değerler:

0=DATA Scientist, 1=Database Administrator, 2=Network Engineer, 3=Software Developer, 4=Tech Support, 5=UI/UX, 6=Web Developer

Temel İstatistik Özeti:



**Count (Sayı):** Her bir sütundaki non-null (null olmayan) değerlerin sayısını temsil eder.

**Mean (Ortalama):** Her bir sütundaki sayısal değerlerin ortalamasını gösterir.

**Std (Standart Sapma):**Her bir sütundaki sayısal değerlerin yayılma (varyans) derecesini ölçer. Daha yüksek bir standart sapma, değerlerin ortalamadan daha fazla dağıldığını gösterir

**25%, 50%, 75% (Çeyrekler):**Verileri küçükten büyüğe sıraladığınızda, %25, %50 ve %75'inci pozisyonlardaki değerleri temsil eder.

**Min (Minimum):**Her bir sütundaki en küçük değeri gösterir.

**Max (Maksimum):**Her bir sütundaki en büyük değeri gösterir.

5.KORELASYON GRAFİĞİ



Korelasyon grafiği ortadan çapraz olarak 2’ye bölünmüştür. Farklı renk paleti ile renkler farklı anlamlara gelebilir. Bu grafikte **mor renkler negatif korelasyonu**,**turuncu renkler pozitif korelasyonu** temsil eder.

**Çok güçlü korelasyona (0.8 ile 1) sahip değişken çiftlerine bakalım:**

**“OS” ile “DSA”=(0.8) “Aptitute” ile “CN”=(0.83)**

**“OS” ile “DBMS”=(0.82) “Aptitute” ile “Mathmatics”=(0.86)**

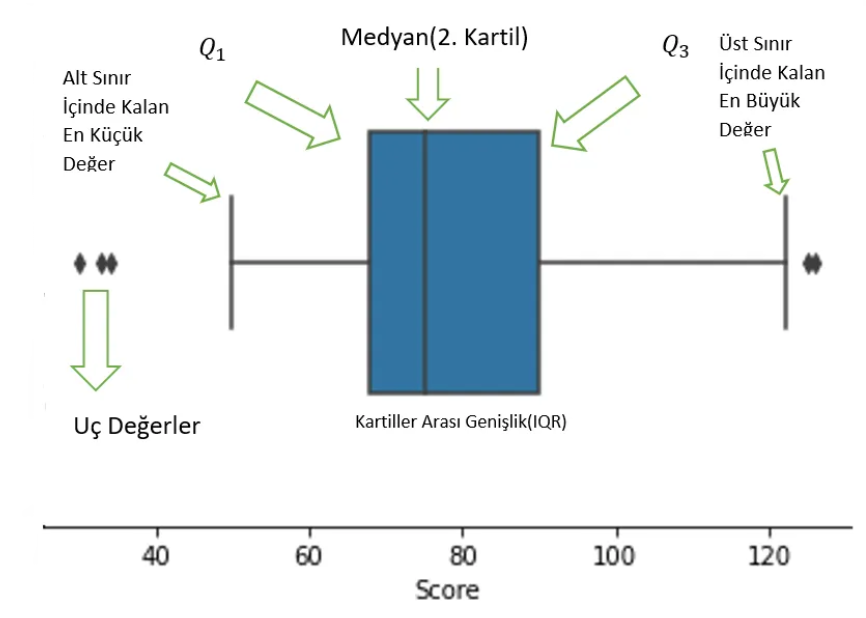
**“CN” ile “DSA”=(0.85) “Problem Solving” ile “Mathmatics”=(0.8)**

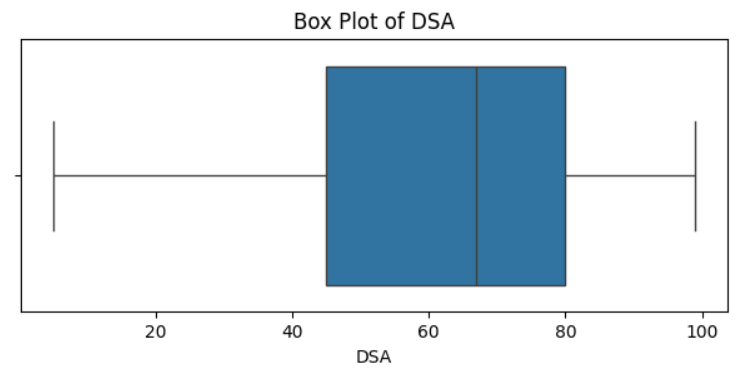
**“Mathmatics” ile “DSA”=(0.85) “Problem Solving” ile “Aptitute”=(0.81)**

**“Aptitute” ile ”DSA”=(0.84)**

6.KUTU-BIYIK GRAFİĞİ

Örnek:

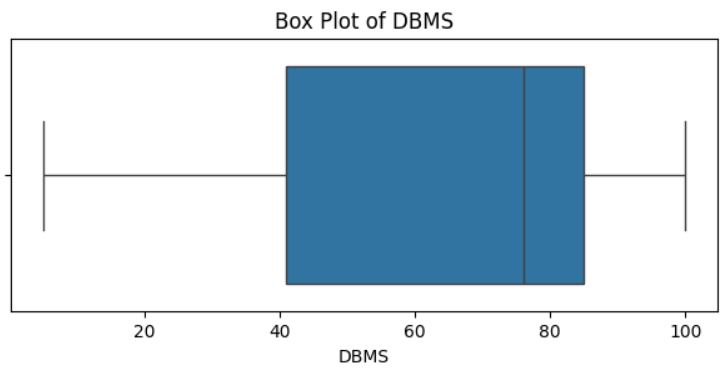


DSA sütunun kutu bıyık grafiğinde alt sınır=5, üst sınır=99

1.Çeyreklik=45

Medyan= 67

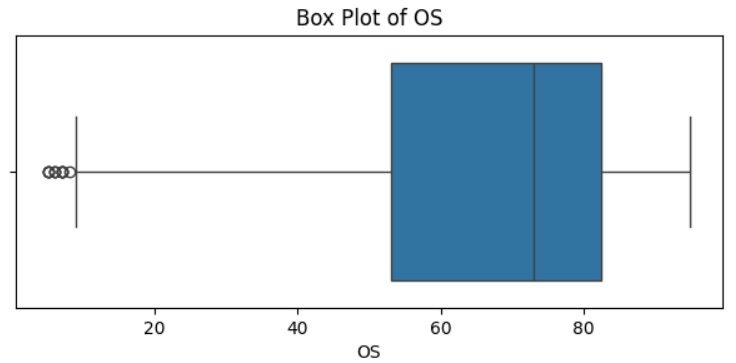
3.Çeyreklik=80

DBMS sütunun kutu bıyık grafiğinde alt sınır=5, üst sınır=100

1.Çeyreklik=40

Medyan= 76

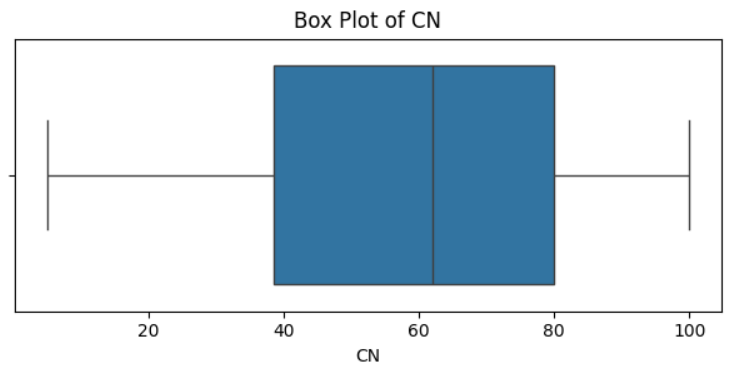
3.Çeyreklik=80

 OS sütunun kutu bıyık grafiğinde alt sınır=5, üst sınır=95

1.Çeyreklik=53

Medyan= 73

3.Çeyreklik=82.5

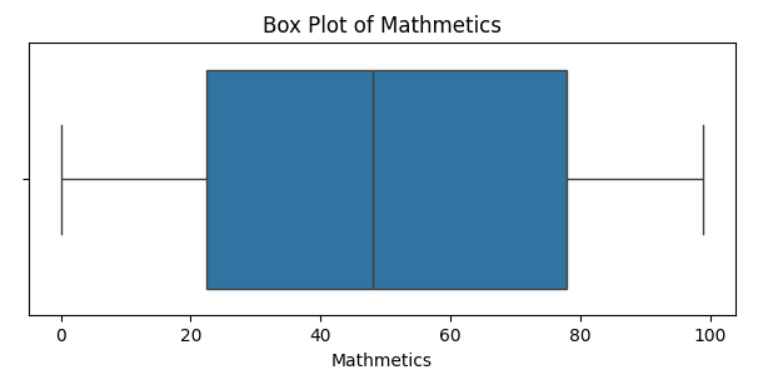


CN sütunun kutu bıyık grafiğinde alt sınır=5, üst sınır=100

1.Çeyreklik=38

Medyan= 62

3.Çeyreklik=80

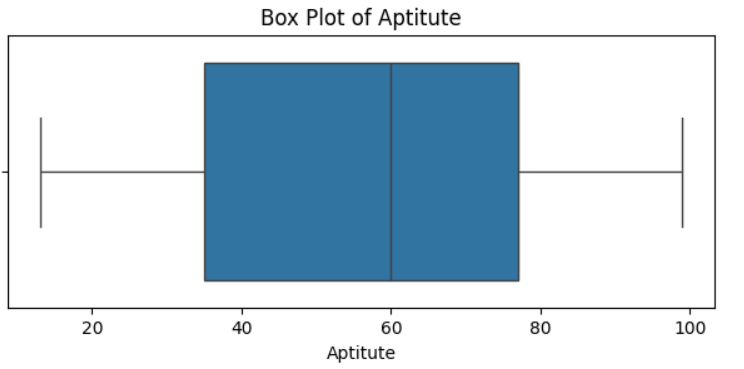


Mathmetics sütunun kutu bıyık grafiğinde alt sınır=0, üst sınır=99

1.Çeyreklik=22

Medyan= 48

3.Çeyreklik=78

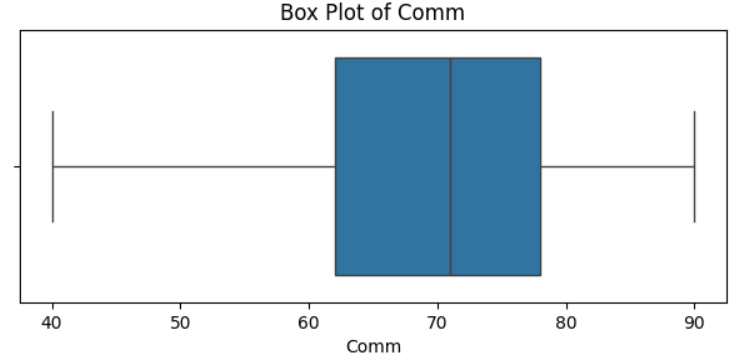


Aptitute sütunun kutu bıyık grafiğinde alt sınır=13, üst sınır=99

1.Çeyreklik=35

Medyan= 60

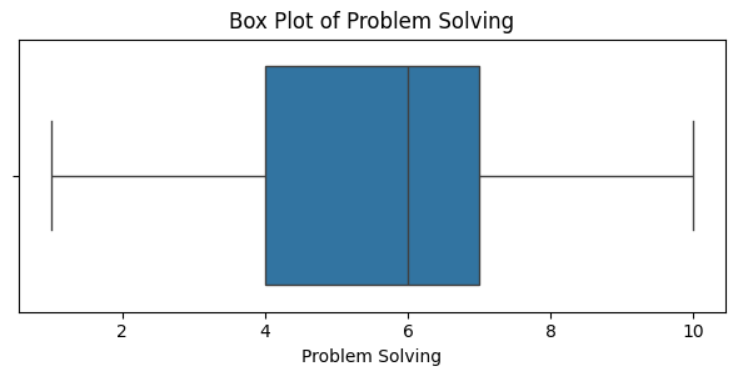
3.Çeyreklik=35

 Comm sütunun kutu bıyık grafiğinde alt sınır=40, üst sınır=90

1.Çeyreklik=62

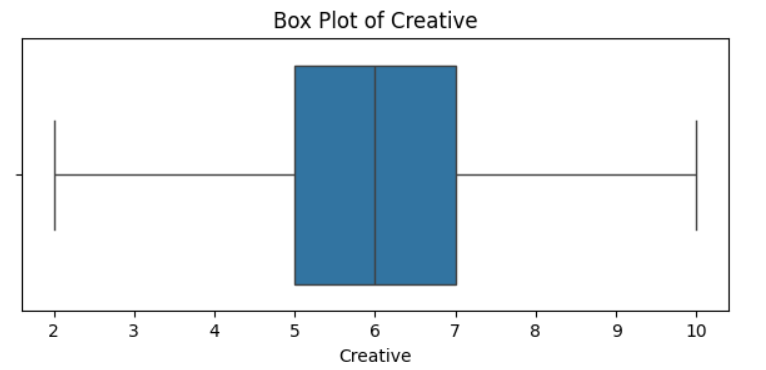
Medyan= 71

3.Çeyreklik=78

Problem Solving sütunun kutu bıyık grafiğinde alt sınır=1, üst sınır=10

1.Çeyreklik=4

Medyan= 6

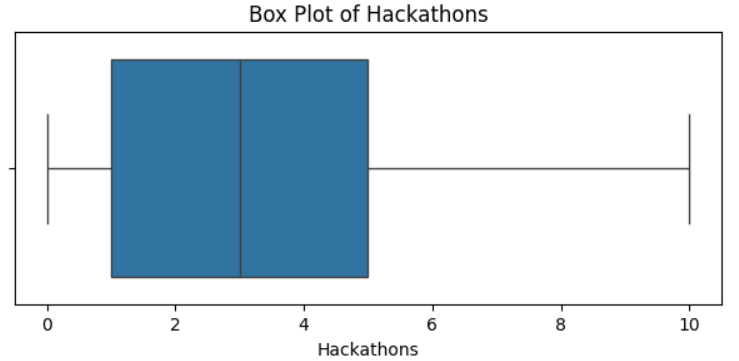
3.Çeyreklik=7

Creative sütunun kutu bıyık grafiğinde alt sınır=2, üst sınır=10

1.Çeyreklik=5

Medyan= 6

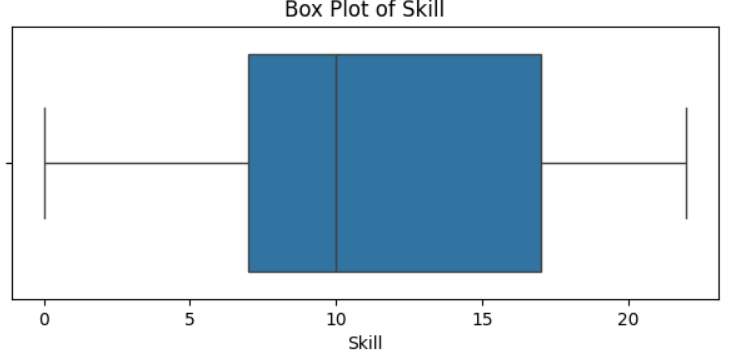
3.Çeyreklik=7

Hackathons sütunun kutu bıyık grafiğinde alt sınır=1, üst sınır=10

1.Çeyreklik=1

Medyan= 3

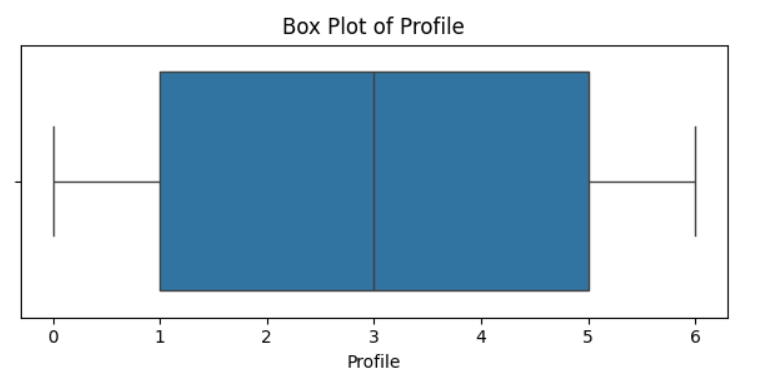
3.Çeyreklik=5

Skill sütunun kutu bıyık grafiğinde alt sınır=0, üst sınır=22

1.Çeyreklik=7

Medyan= 10

3.Çeyreklik=17

Profile sütunun kutu bıyık grafiğinde alt sınır=0, üst sınır=

1.Çeyreklik=1

Medyan= 3

3.Çeyreklik=5

7.Histogram Grafikleri

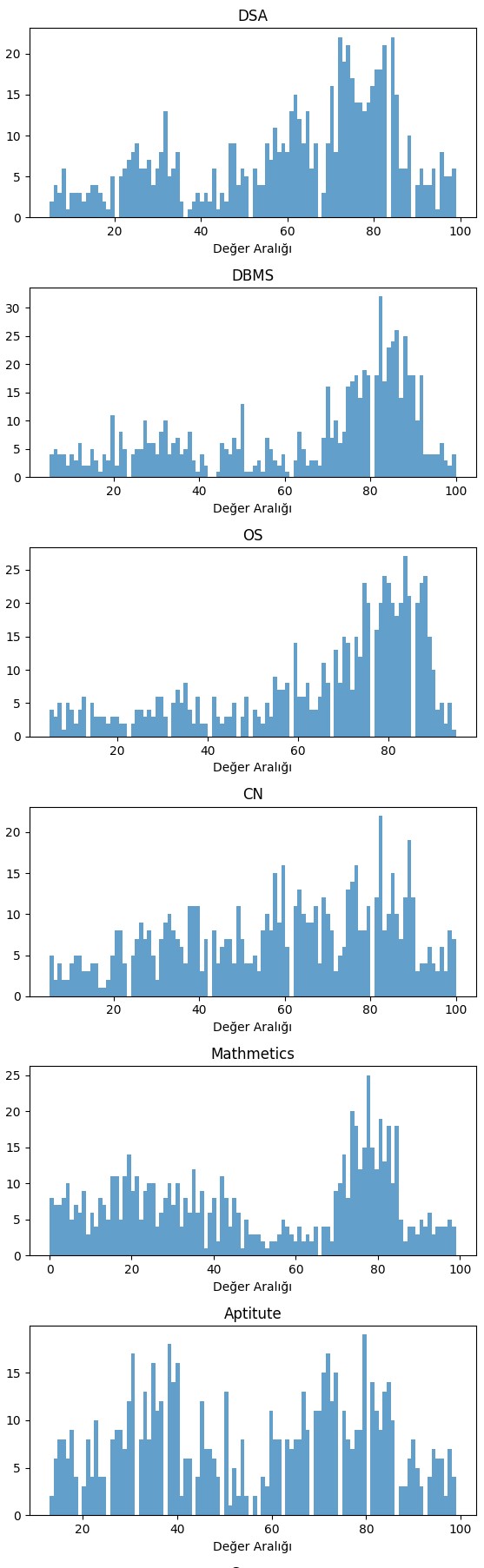
Histogram, bir veri setinin dağılımını gösteren bir grafik türüdür. Genellikle sürekli bir değişkenin değerlerini belirli aralıklara bölen ve her aralıktaki gözlemlerin sayısını gösteren sütunlu bir grafiktir. Bu grafik, veri setinin genel formunu, merkezi eğilimini, dağılımını ve aykırı değerleri görselleştirmek için kullanılır.

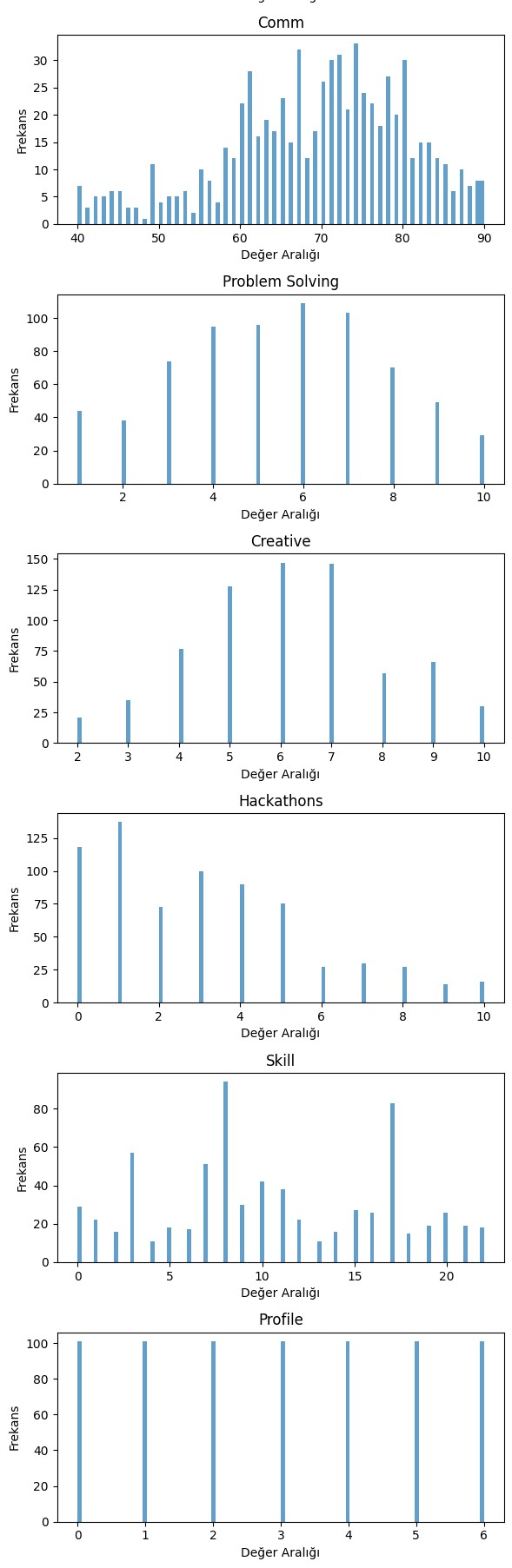
Histogramın temel özellikleri şunlardır:

1. **Sütunlar (Barlar):** Histogram, belirli aralıklarda bulunan değerlerin sayısını temsil eden dikdörtgen sütunlardan oluşur. Her sütun, bir aralıktaki değerlerin frekansını ifade eder.
2. **Aralıklar (Bins):** Veri seti değerlerini bölmek için kullanılan aralıklar histogramdaki sütunları belirler. Genellikle bu aralıklar eşit büyüklükte olup, veri setindeki değerlerin geniş bir yelpazede görüntülenmesini sağlar.
3. **Eksenler:** Histogramın yatay ekseninde değerlerin aralıkları, dikey ekseninde ise her aralıktaki değerlerin frekansları yer alır.

Histogramın yorumlanması aşağıdaki adımları içerir:

* **Merkezi Eğilim:** Histogram, veri setindeki merkezi eğilimi görselleştirmek için kullanılabilir. Veri setindeki modu (en sık tekrarlanan değer), medyanı veya ortalama değeri belirleme konusunda yardımcı olabilir.
* **Dağılım:** Histogram, veri setindeki değerlerin nasıl dağıldığını gösterir. Normal dağılmış bir veri seti, simetrik bir histograma sahip olabilirken, çarpık bir dağılım farklı bir şekilde görünebilir.
* **Aykırı Değerler:** Histogram, aykırı değerleri tanımlamak için kullanılabilir. Eğer bir aralıkta diğerlerine göre çok daha fazla gözlem varsa veya bir sütun diğerlerine göre belirgin bir şekilde uzunsa, bu durumda aykırı değerler olabilir.





8.KODLAR



