AF-SM-04\_05.2020.G

|  |  |
| --- | --- |
| **კურსის სახელწოდება:** | **მანქანური სწავლების საწყისები** |
| **ლექტორი:** | **დავით დათუაშვილი** |

ვარიანტი 3

მაქსიმალური ქულა 30

1. მოცემულია ფაილი heart.csv, რომელშიც target ცვლადი არის დამოკიდებული ცვლადი(ადამიანს აქვს თუ არა გულის პრობლემა), დანარჩენი ყველა კი არის დამოუკიდებელი ცვლადები .უპასუხეთ ქვევით მოცემულ კითხვებს(10 ქულა)

* განაცალკევეთ დამოკიდებული და დამოუკიდებელი ცვლადები ერთმანეთისგან, დაყავით სატრენინგო და სატესტო ნაწილებად(სატესტო ზომა იყოს 25%(2 ქულა)
* import pandas as pd  
  from sklearn.ensemble import RandomForestClassifier  
  from sklearn.model\_selection import train\_test\_split  
    
  data = pd.read\_csv("heart.csv")  
    
  target = data["target"].values  
  X = data.drop("target", axis=1).values  
    
  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, target, test\_size=0.25, random\_state=5)
* გამოიყენეთ მეთოდი RandomForestClassifier (sklearn.ensemble ბიბლიოთეკიდან), დაატრენინგეთ და გამოთვალეთ მისი score სატესტო მნიშვნელობაზე, აჩვენეთ თუ რამდენად დიდი სხვაობაა მათ შორის (4 ქულა)

import pandas as pd  
from sklearn.ensemble import RandomForestClassifier  
from sklearn.model\_selection import train\_test\_split  
  
data = pd.read\_csv("heart.csv")  
  
target = data["target"].values  
X = data.drop("target", axis=1).values  
  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, target, test\_size=0.25, random\_state=5)  
modelForTest = RandomForestClassifier()  
modelForTrain = RandomForestClassifier()  
modelForTest.fit(X\_test, y\_test)  
modelForTrain.fit(X\_train, y\_train)  
  
print(modelForTest.score(X\_test, y\_test))  
print(modelForTrain.score(X\_train, y\_train))

* წინა მონაცემებზე დაყრდნობით PCA ის მეთოდის მეშვეობით შეამცირეთ განზომილება 6 სვეტამდე და გამოიყენეთ RandomForestClassifier ის ალგორითმი გამოთვალეთ როგორც სატრენინგო , ასევე სატესტო score და შეადარეთ წინა მონაცემებს თუ რამდენად იმოქმედა განხომილების შემცირებამ სიზუსტეზე.ასევე გამოიყენეთ criterion =“entropy” პარამეტრი და აღწერეთ მისი შედეგი სიზუსტეზე.(4 ქულა)
* import pandas as pd  
  from sklearn.ensemble import RandomForestClassifier  
  from sklearn.model\_selection import train\_test\_split  
  from sklearn.linear\_model import LinearRegression  
  from sklearn.decomposition import PCA  
    
  data = pd.read\_csv("heart.csv")  
    
  y = data["target"].values  
  X = data.drop("target", axis=1).values  
    
  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.25, random\_state=5)  
  PCA = PCA(n\_components=6)  
  Xnew = PCA.fit\_transform(X)  
    
  modelForTest = RandomForestClassifier()  
  modelForTrain = RandomForestClassifier()  
  modelForXnew = RandomForestClassifier(criterion="entropy")  
    
  modelForTest.fit(X\_test, y\_test)  
  modelForTrain.fit(X\_train, y\_train)  
  modelForXnew.fit(Xnew, y)  
    
  print(modelForTest.score(X\_test, y\_test))  
  print(modelForTrain.score(X\_train, y\_train))  
  print(modelForXnew.score(Xnew, y))

Sedegi miviRe zustad igive.

1. მოცემულია ფაილი Cars.csv, რომელშიც price არის დამოკიდებული ცვლადი , ხოლო ყველა სხვა ცვლადი კი დამოუკიდებელი ცვლადი(მიზანი არის მანქანის ფასის პროგნოზირება სხვადასხვა(პარამეტრებით).უპასუხეთ ქვევით მოცემულ კითხვებს(10 ქულა)

* განაცალკევეთ დამოკიდებული და დამოუკიდებელი ცვლადები ერთმანეთისგან, სატრენინგო ზომად აიღეთ 30%, მოახდინეთ მათი სტანტარდიზირება და PCA ის გამოყენებით მოახდინეთ მონაცემების განზომილების შემცირება 16 ი ცვლადიდან 9 ცვლადამდე(ასევე აჩვენეთ თუ 9 ახალი ცვლადი ინფორმაციის რამდენ პროცენტს ინახავს თავდაპირველი მონაცემებიდან(5 ქულა)
* data2 = pd.read\_csv("Cars.csv")  
  # fueltype = pd.get\_dummies(data2["fueltype"])  
  # aspiration = pd.get\_dummies(data2["aspiration"])  
  # data2 = pd.concat([fueltype, aspiration], axis=1)  
  y2 = data2["price"].values  
  X2 = data2.drop("price", axis=1).values  
    
  X2\_train, X2\_test, y2\_train, y2\_test = train\_test\_split(X2, y2, test\_size=0.3, random\_state=5)
* LInearRegression ის მოდელის მეშვეობით გამოთვალეთ score როგორც სატრენინგო , ასევე სატესტო მონაცემებზე(5 ქულა)

data2 = pd.read\_csv("Cars.csv")  
# fueltype = pd.get\_dummies(data2["fueltype"])  
# aspiration = pd.get\_dummies(data2["aspiration"])  
# data2 = pd.concat([fueltype, aspiration], axis=1)  
y2 = data2["price"].values  
X2 = data2.drop("price", axis=1).values  
  
X2\_train, X2\_test, y2\_train, y2\_test = train\_test\_split(X2, y2, test\_size=0.3, random\_state=5)  
  
model2ForTest = LinearRegression()  
model2ForTrain = LinearRegression()  
model2ForTest.fit(X2\_test, y2\_test)  
model2ForTrain.fit(X2\_train, y2\_train)  
  
print(model2ForTest.score(X2\_test, y2\_test))  
print(model2ForTrain.score(X2\_train, y2\_train))  
print(data2)

1. მოცემულია ფაილი Cluster.csv, გრაფიკული დიაგრამის (scatter chart) ის მეშვეობით განსაზღვრეთ თუ რამდენი კლასტერია მოცემულ მონაცემებში და KMeans ის ალგორითმის მეშვეობით მოათავსეთ ამ კლასტერების ცენტრები დიაგრამზე. ჩასვით დიაგრამა კოდთან ერთად მოცემულ დავალებაში (10 ქულა)
2. data3 = pd.read\_csv("Cluster.csv")  
     
   X3 = data3["Feature1"].values  
   y3 = data3["Feature2"].values  
   scatter = scatter(X3, y3)  
   true\_k = 2  
   model = KMeans(n\_clusters=true\_k, init='k-means++', max\_iter=100, n\_init=1)  
   model.fit(scatter)

ეს ბოლო არ მისწავლია და რაღაც საშინელება დავწერე :D