QUANTITTIVE III LAB

BCADA3P3

RAISA

20BCAB09

LAB 1

#LINEAR ALGEBRA

Created on Fri Oct 1 11:31:09 2021

@author: Raisa

"""

```
import numpy as np
#1)create and print 7 tuple 1 D array
x=(3,5,8,1,4,9,78)
print(x)
#2)print 2nd 4th 7th elements of your array
print(x[1])
print(x[3])
print(x[6])
#3)shape of array
print(np.shape(x))
#4)tranpose of array
X=np.array([[3,5,8,1,4,9,78]]).T
print(X)
print(np.shape(X))
#5)4x5 vector matrix
```

```
z=np.array([[1,1,2,3,2],[22,3,55,1,5],[3,7,8,66,8],[98,2,1,0,7]])
print(z)
#6)shape of array
print(np.shape(z))
#7)tranpose of array
print(z.T)
Z=np.array([[1,1,2,3,2],[22,3,55,1,5],[3,7,8,66,8],[98,2,1,0,7]]).T
print(Z)
#8)first row of matrix
print(z[1,4])
print(z[0])
#9)first col of matrix
print(z[:,0])
#10)7D array with 0 and 3D array with 1
a=np.array([[[[[[[1,2,3]]]]]]])
zer=np.zeros((7,7))
print(zer)
one=np.ones((3,3))
print(one)
print(a)
print(a.ndim)
c=np.array([(1,2,3,4,5,6,7),(1,2,3,4,5,6,7),(1,2,3,4,5,6,7),(1,2,3,4,5,6,7),(1,2,3,4,5,6,7)])
print(c)
print(c.ndim)
print(c.reshape(7,5))
print(c)
b=np.array([[[1,1,1]]]*3)
print(b)
print(b.ndim)
```

LAB 2

LINEAR ALGEBRA 2

Created on Tue Oct 12 10:42:55 2021

@author: Raisa

"""

```
import numpy as np
#1)Define and print a 6 dimentional vector
a=np.array([[[[[[1,2,3]]]]]])
print(a)
print(a.ndim)
#2)Print the transpose of the above vector
print(a.T)
#3)Define two non square matrices such that they can be mulplied.
x=np.array([[3,4]])
y=np.array([[2],[4]])
print(x)
print(y)
#4) Print the shape of the above matrices
print(x.shape)
print(y.shape)
#5)Print the product of above two matrices (do so without using the inbuiltfunctions).
z=np.array([[0]])
for i in range(len(x)):
    for j in range(len(y[0])):
        for k in range(len(y)):
            z[i][j]+=x[i][k]*y[k][j]
for r in z:
```

```
   print(r)
#6)Define two non square matrices of same order and print their sum.
p=np.array([[2,4,3,4],[1,2,3,4]])
q=np.array([[4,5,6,7],[4,3,2,1]])
print(p)
print(q)
print(p+q)
#7)Define a square matrix A.
A=np.array([[1,2,3],[7,3,9],[3,8,10]])
print(A)
#8) Print the transpose of A.
print(A.T)
#9)Print the identity matrix of the above order I.
I=np.identity(3,int)
print(I)
#10) Verify A.I = I.A for matrix multiplication.
print(np.dot(A,I))#A.I
print(np.dot(I,A))#I.A
#therefore A.I = I.A
#11) Define another square matrix of the same order as A
B=np.array([[5,6,0],[1,4,1],[0,2,1]])
print(B)
#12)Print the product of the matrices as matrix multiplication
print(np.dot(A,B))
#13) Print the product of the matrices by element wise multiplication
print(4*B)
print(A*B)
#14)  Calculate and print the inverse of A. (Use linalg)
#Check if determinant is 0
```

```
det=print(np.linalg.det(A))
#Use if else statement to calculate inverse only when determinant is non 0
if det!=0:
   print('inverse exists')
   print(np.linalg.inv(A))
else:
   print('inverse does not exixts')
```

LAB 3

VISUALIZATION

Created on Fri Oct 22 14:09:39 2021

@author: Raisa

"""

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import csv
mydata=pd.read_csv("http://winterolympicsmedals.com/medals.csv")
#mydata.info()
#print(mydata.head())
#print(mydata.describe())
#sns.countplot(x='City',data=mydata)
#sns.countplot(x='Sport',data=mydata)
#sns.countplot(x='Discipline',data=mydata)
#sns.countplot(x='Event gender',data=mydata)
#sns.countplot(x='Medal',data=mydata)
```

```
#sns.countplot(x='NOC',data=mydata)
#sns.countplot(x='Year',data=mydata)
#sns.countplot(x='Event',data=mydata)
#sns.scatterplot('Sport','Discipline',data=mydata)
#sns.boxplot(y="Year",x="Event gender",data=mydata)
#sns.distplot(mydata["Year"])
#sns.jointplot(mydata["Year"],mydata["Sport"])
#sns.pointplot(mydata['Year'],mydata['Sport'],hue=mydata['Event gender'])
#plt.show()
```

LAB 4

EDA AND LINEAR REGRESSION

Created on Tue Nov 9 16:35:19 2021

```
@author: Raisa
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import linear_model
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
from sklearn.model_selection import train_test_split
mydata=pd.read_csv("/Users/Raisa/PycharmProjects/stats_eda/heart.csv")
print(mydata.shape)
print(mydata.head(10))
mydata.info()
print(mydata.describe())
print(mydata.dtypes)
```

```
#since fastingbp ang heartdisease are categorical values we convert that to str data type
mydata['FastingBS'] = mydata['FastingBS'].astype(str)
mydata['HeartDisease'] = mydata['HeartDisease'].astype(str)
df_desc=pd.DataFrame(mydata.describe())
#df_desc.to_csv("/Users/Raisa/PycharmProjects/stats_eda/describe.csv")
# Do we have duplicates?
print('Number of Duplicates:', len(mydata[mydata.duplicated()]))
# Do we have missing values?
print('Number of Missing Values:', mydata.isnull().sum().sum())
corr_ht=mydata.corr()
print(corr_ht)
sns.heatmap(corr_ht)
print(mydata.nunique())
print(mydata['ChestPainType'].unique())
print(mydata['ST_Slope'].unique())
print(mydata['RestingECG'].unique())
sns.pairplot(mydata)
sns.relplot(x="Oldpeak",y="RestingBP",hue="Sex",data=mydata)
X=mydata[['MaxHR']]
Y=mydata['Age']
#X_train, X_test,Y_train,Y_test = train_test_split(x,y,test_size =0.2)
X_train = X[:-20]
X_test = X[-20:]
Y_train = Y[:-20]
Y_test = Y[-20:]

reg = linear_model.LinearRegression()
reg.fit(X_train,Y_train)
print(reg.coef_)
```

```
A=r2_score(X,Y)
print(A)
B=mean_squared_error(X,Y)
print(B)
```

LAB 5

MULTIPLE REGRESSION

```
#multiple regression
X=mydata[['Age']]
Y=mydata[['RestingBP','Oldpeak']]
#X_train, X_test,Y_train,Y_test = train_test_split(x,y,test_size =0.2)
X_train = X[:-20]
X_test = X[-20:]
Y_train = Y[:-20]
Y_test = Y[-20:]

reg = linear_model.LinearRegression()
reg.fit(X_train,Y_train)
print(reg.coef_)
A=r2_score(X,Y)
print(A)
B=mean_squared_error(X,Y)
print(B)
```

LAB 6

HYPOTHESIS TESTING IN R

```
getwd()
```

```
setwd("C:/Users/Raisa/Documents/jerry(DS)/datasets")

data=read.csv("heart.csv")

t.test(data$RestingBP,mu=30)

t.test(data$RestingBP,mu=132 ,alternative="less",conf.level=0.85)

mean(data$RestingBP)


#if p value less than alpha(significance value = 1 - confidence level) reject the null hyposthesis

#Ho: mean = 30

#H1: mean is not equal to 30

#attach(CO2)

#View(CO2)

#t.test(uptake,mu=30)# if mu=27.2131, p-value=1

# below is one tailed test whereas above one is two tailed

# perform one sample T test ( one tailed)

# Ho: mean is 30

# H1: mean is less than 30

#t.test(uptake,mu=30,alternative="less",conf.level=0.95)

# Performs independent sample t test

# Ho: mean uptake of CO2 of quebec= mean uptake of CO2 of mississippi

# H1: mean uptake of CO2 of quebec!= mean uptake of CO2 of mississippi

#t.test(uptake~Type,mu=0,alt='two.sided',conf=0.95,paired=FALSE)
```

LAB 7

FACTOR ANALYSIS

Created on Wed Jan 5 19:30:25 2022

@author: Raisa

"""

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.decomposition import FactorAnalysis
from sklearn import preprocessing
from sklearn.datasets import load_digits
mydata=pd.read_csv("C:/Users/Raisa/Documents/jerry(DS)/datasets/Walmart_Store_sales.csv")
print(mydata.drop('Date',axis=1,inplace=True))
print(mydata.shape)
print(mydata.head(10))
mydata.info()
print(mydata.describe())
print(mydata.dtypes)
data_normal = preprocessing.scale(mydata)
fa = FactorAnalysis(n_components = 1)
fa.fit(data_normal)
for score in fa.score_samples(data_normal):
    print(score)
```

LAB 8

LOGISTIC REGRESSION

Created on Thu Jan 13 15:17:44 2022

@author: Raisa

"""

```
from sklearn.datasets import make_classification
from matplotlib import pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
import pandas as pd
mydata=pd.read_csv("C:/Users/Raisa/Documents/jerry(DS)/datasets/salary_data.csv")
print(mydata.shape)
print(mydata.describe())
print(mydata.dtypes)
print(mydata.value_counts())
# Do we have missing values?
print('Number of Missing Values:', mydata.isnull().sum().sum())
left = mydata[ mydata['left'] == 1]
print(left.shape)
retained = mydata[mydata['left'] == 0]
print(retained.shape)
data=pd.DataFrame(mydata.groupby('left').count())
#sns.countplot(x="left",data=mydata)
#sns.countplot(x="left",hue="salary",data=mydata)
#corr_ht=mydata.corr()
#print(corr_ht)
#sns.heatmap(corr_ht)
sns.countplot(x="salary",hue="left",data=mydata)
sns.countplot(x="Department",hue="left",data=mydata)
sns.countplot(x="satisfaction_level",hue="left",data=mydata)
```

```
sns.countplot(x="promotion_last_5years",hue="left",data=mydata)

newd=mydata[["left","satisfaction_level","salary","promotion_last_5years"]]


X_train,X_test, y_train, y_test
=train_test_split(newd[["left","satisfaction_level","salary","promotion_last_5years"]],mydat
a.left,test_size=0.3, stratify=newd['left'])

model=LogisticRegression()

model.fit(X_train,y_train)

model.predict(X_test)

model.score(X_test, y_test)

# Create a scatter plot

#scatter(x, y, c=y, cmap='rainbow')

#plt.title('Scatter Plot of Logistic Regression')

#plt.show()
```

LAB 9

CLUSTER ANALYSIS(HIERARCHICAL)

```
Created on Sun Jan 30 19:54:27 2022


@author: Raisa
"""


from sklearn.cluster import AgglomerativeClustering

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from scipy.cluster.hierarchy import dendrogram,linkage


df=pd.read_csv("/Users/Raisa/PycharmProjects/stats_eda/heart.csv")
```

```
#df = mydata.mydata
#df = mydata[ :, : ]
Z = linkage(df, method = "ward")
dendro = dendrogram(Z)
plt. title( 'Dendogram' )
plt.ylabel( 'Euclidean distance')
plt. show()

ac = AgglomerativeClustering(n_clusters=2,affinity="euclidean", linkage="ward")
labels = ac. fit_predict (df)
plt. figure(figsize = (8,5) )
plt.scatter (df[labels == 0 ,0], df[labels == 0, 1] , c = 'red' )
plt.scatter (df [labels == 1 , 0] , df [labels == 1,1] , c = 'blue')
plt.scatter (df [labels == 2 , 0], df [labels == 2 , 1],c = 'green' )
plt.scatter (df [labels == 3 , 0], df [ labels == 3 ,1],c = 'black')
plt.scatter (df[labels == 4 ,] , df [ labels == 4 , 1], c = 'orange' )
plt. show()
```