Name: Allan Rodrigues
Class: TE IT A
Roll no: 59
Pid:191104

St. Francis Institute of Technology, Mumbai-400 103

Department of Information Technology

A.Y. 2021-2022

Class: TE-ITA/B, Semester: VI

Subject: **Data Science Lab**

Experiment – 3: To implement Data Modelling.

1. **Aim:** To implement Data Modelling
2. **Objectives:** After study of this experiment, the student will be able to
   - Understand how data to be re-scaled
   - Understand how data partitioning in done using sklearn
3. **Outcomes:** After study of this experiment, the student will be able to
   - Understand data rescaling and data partitioning using sklearn
4. **Prerequisite:** Fundamentals of Python Programming and Database Management System.
5. **Requirements:** Python Installation, Personal Computer, Windows operating system, Internet Connection, Microsoft Word.
6. **Pre-Experiment Exercise:**

   Brief Theory:
   - Concept of sklearn package for machine learning**.**

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. Extensions or modules for SciPy care conventionally named SciKits. As such, the module provides learning algorithms and is named scikit-learn. The library is focused on modeling data. It is not focused on loading, manipulating and summarizing data. For these features, refer to NumPy and Pandas.

Laboratory Exercise

A. **Procedure:** (the sheet for commands in attached with the file)

B. Paste Screenshots of above commands.

Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

```
[1] import matplotlib.pyplot as plt
    import seaborn as sns
    import pandas as pd
    import numpy as np

    sns.get_dataset_names()
```

```
['anagrams',
 'anscombe',
 'attention',
 'brain_networks',
 'car_crashes',
 'diamonds',
 'dots',
 'exercise',
 'flights',
 'fmri',
 'gammas',
 'geyser',
 'iris',
 'mpg',
 'penguins',
 'planets',
 'taxis',
 'tips',
```

Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

```
[2] df= sns.load_dataset("titanic")
    df
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S	Second	man	True	NaN	Southampton	no	True
887	1	1	female	19.0	0	0	30.0000	S	First	woman	False	B	Southampton	yes	True
888	0	3	female	NaN	1	2	23.4500	S	Third	woman	False	NaN	Southampton	no	False
889	1	1	male	26.0	0	0	30.0000	C	First	man	True	C	Cherbourg	yes	True
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	True	NaN	Queenstown	no	True

Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

```
[3] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   survived     891 non-null    int64
 1   pclass       891 non-null    int64
 2   sex          891 non-null    object
 3   age          714 non-null    float64
 4   sibsp        891 non-null    int64
 5   parch        891 non-null    int64
 6   fare         891 non-null    float64
 7   embarked     889 non-null    object
 8   class        891 non-null    category
 9   who          891 non-null    object
 10  adult_male   891 non-null    bool
 11  deck         203 non-null    category
 12  embark_town  889 non-null    object
 13  alive        891 non-null    object
 14  alone        891 non-null    bool
dtypes: bool(2), category(2), float64(2), int64(4), object(5)
memory usage: 80.7+ KB
```

Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

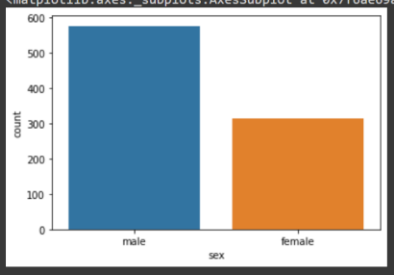
Comment Share

+ Code + Text

RAM Disk Editing

```
[4] #count plot: categorical data
    sns.countplot(x="sex",data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6ae69a3190>
```




```
[5] df.groupby("sex")["survived"].value_counts()
```

```
sex     survived
female  1           233
        0            81
male    0           468
```

Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

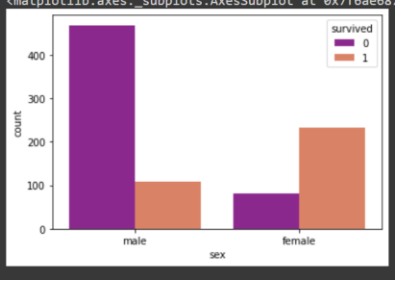
+ Code + Text

RAM Disk Editing

```
[5] sex     survived
    female  1           233
            0            81
    male    0           468
            1           109
    Name: survived, dtype: int64
```

```
[6] sns.countplot(x="sex",hue="survived",data=df,palette="plasma")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6ae6877790>
```



Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

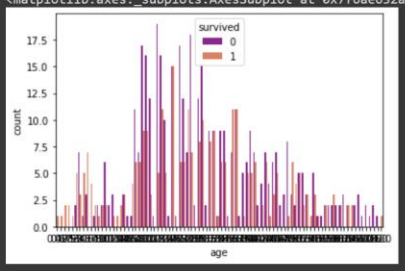
Comment Share

+ Code + Text

RAM Disk Editing

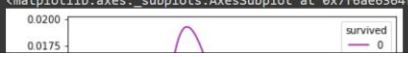
```
[7] sns.countplot(x="age",hue="survived",data=df,palette="plasma")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6ae632aad0>
```




```
[8] #KDE Plot:  Kernel density estimate plot: show distribution of continiuous data
    sns.kdeplot(x="age",hue="survived",data=df,palette="plasma")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6ae6364fd0>
```



Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

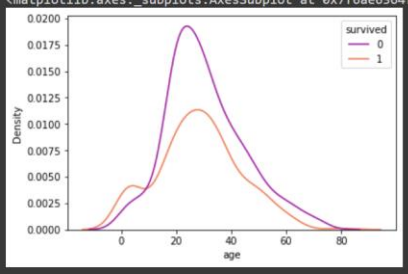
Comment Share

+ Code + Text

RAM Disk Editing

[8] #KDE Plot: Kernel density estimate plot: show distribution of continiuous data
sns.kdeplot(x="age",hue="survived",data=df,palette="plasma")

<matplotlib.axes._subplots.AxesSubplot at 0x7f6ae6364fd0>




```
[9] # Distribution plot --> similar kde --> probability across other values
    sns.displot(x="age",hue="survived",data=df,bins=5,kde=True)
```

Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

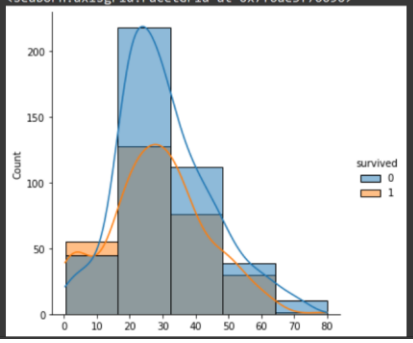
Comment Share

+ Code + Text

RAM Disk Editing

[9] # Distribution plot --> similar kde --> probability across other values
sns.displot(x="age",hue="survived",data=df,bins=5,kde=True)

```
<seaborn.axisgrid.FacetGrid at 0x7f6ae3f70090>
```



Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

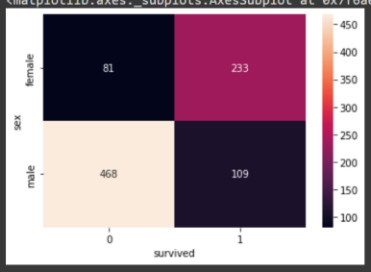
+ Code + Text

RAM Disk Editing

```
[10] #heat Map:
     group = df.groupby(['sex','survived'])
     class_survived = group.size().unstack()

     sns.heatmap(class_survived,annot=True,fmt="d")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6ae3ee49d0>
```



Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

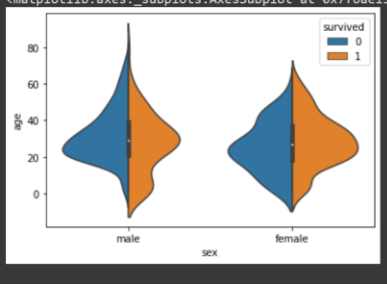
Comment Share

+ Code + Text

RAM Disk Editing


```
[11] #violin plot
     sns.violinplot(x="sex",y="age",hue="survived",data=df,split=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6ae15e7210>
```



Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

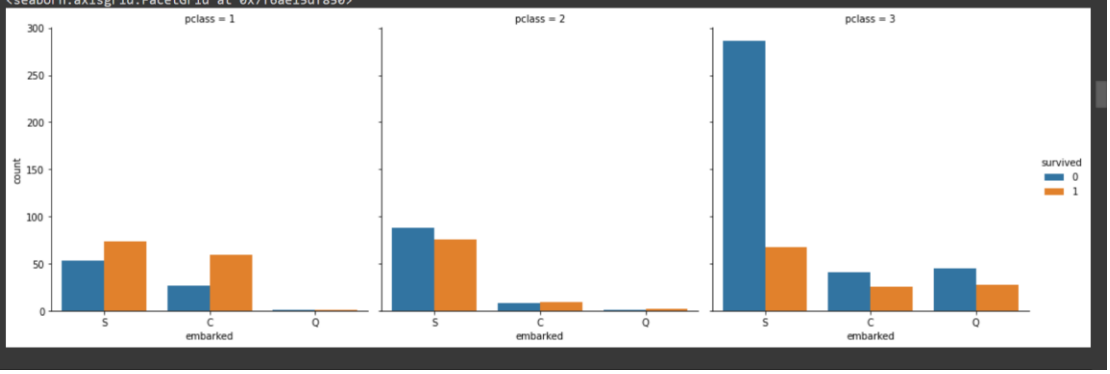
Comment Share

+ Code + Text

RAM Disk Editing

```
[12] #Catplot
     sns.catplot(x="embarked",kind="count",col="pclass",hue="survived",data=df)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f6ae15df850>
```



Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

```
[22] import seaborn as sns
     import matplotlib.pyplot as plt
     df=sns.load_dataset('iris')
     print(df)
```

```
     sepal_length  sepal_width  petal_length  petal_width    species
0             5.1          3.5           1.4          0.2     setosa
1             4.9          3.0           1.4          0.2     setosa
2             4.7          3.2           1.3          0.2     setosa
3             4.6          3.1           1.5          0.2     setosa
4             5.0          3.6           1.4          0.2     setosa
..            ...          ...           ...          ...        ...
145           6.7          3.0           5.2          2.3  virginica
146           6.3          2.5           5.0          1.9  virginica
147           6.5          3.0           5.2          2.0  virginica
148           6.2          3.4           5.4          2.3  virginica
149           5.9          3.0           5.1          1.8  virginica

[150 rows x 5 columns]
```

Exp 3.ipynb

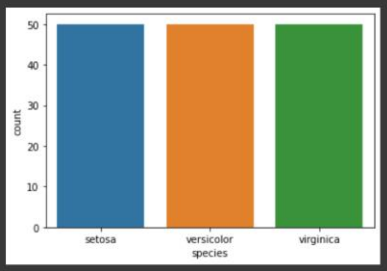
File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

```
[23] sns.countplot(x='species',data=df)
     plt.show()
```




```
sns.scatterplot(x='sepal_length',y='petal_length',data=df,hue='species')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6ae1474490>
```

Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

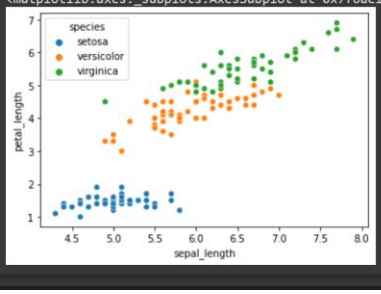
Comment Share

+ Code + Text

RAM Disk Editing

```
sns.scatterplot(x='sepal_length',y='petal_length',data=df,hue='species')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6ae1474490>
```



Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

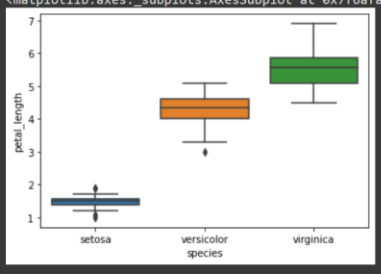
Comment Share

+ Code + Text

RAM Disk Editing


```
[16] sns.boxplot(x='species',y='petal_length',data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6afab825d0>
```



Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

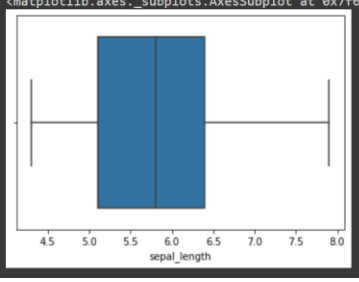
Comment Share

+ Code + Text

RAM Disk Editing

```
[17] sns.boxplot(x='sepal_length',data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6ae12ad550>
```



Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

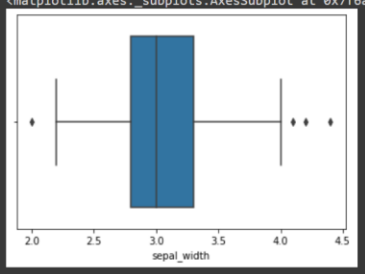
Comment Share

+ Code + Text

RAM Disk Editing

```
[18] sns.boxplot(x='sepal_width',data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6ae1225310>
```




```
[19] # IQR- Interquartile range- Divide data into four equal parts.
     # Q1- Middle number between the smallest no and the median of the dataset.
     # Q2- Median of the data.
     # Q3- Middle values between the median and the highest value of the dataset.
```

Exp 3.ipynb

File Edit View Insert Runtime Tools Help

Comment Share

+ Code + Text

RAM Disk Editing

```
import sklearn
from sklearn.datasets import load_boston
import pandas as pd
import seaborn as sns
import numpy as np

df=pd.read_csv("https://raw.githubusercontent.com/uiuc-cse/data-fa14/gh-pages/data/iris.csv")

Q1 = np.percentile(df['sepal_width'],25,interpolation='midpoint')
Q3 = np.percentile(df['sepal_width'],75,interpolation='midpoint')
IQR= Q3 - Q1
print("old shape",df.shape)

#upper bound
upper=np.where(df['sepal_width']) >= (Q3+1.5*IQR)
lower=np.where(df['sepal_width']) <= (Q1-1.5*IQR)

df.drop(upper[0],inplace = True)
df.drop(lower[0],inplace = True)

print("new shape",df.shape)
sns.boxplot(x='sepal_width',data=df)
```

```
old shape (150, 5)
```

Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

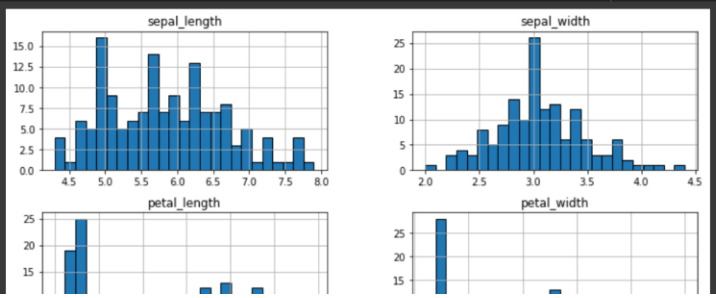
+ Code + Text

RAM Disk Editing

[28]

```
import sklearn
from sklearn.datasets import load_boston
import pandas as pd
import seaborn as sns
import numpy as np

df=pd.read_csv("https://raw.githubusercontent.com/uiuc-cse/data-fa14/gh-pages/data/iris.csv")
df.hist(edgecolor='black',bins=25,figsize=(12,6))
plt.show()
```



Exp 3.ipynb

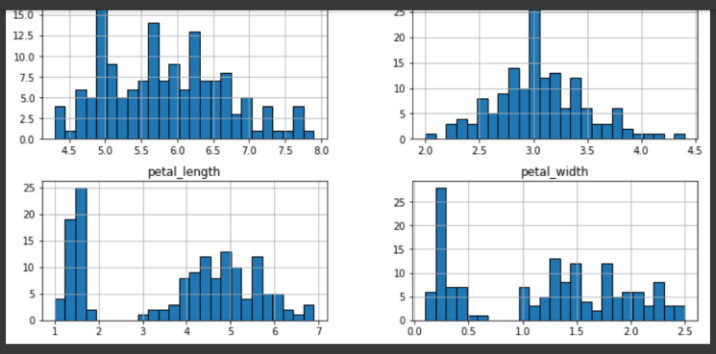
File Edit View Insert Runtime Tools Help

Comment Share

+ Code + Text

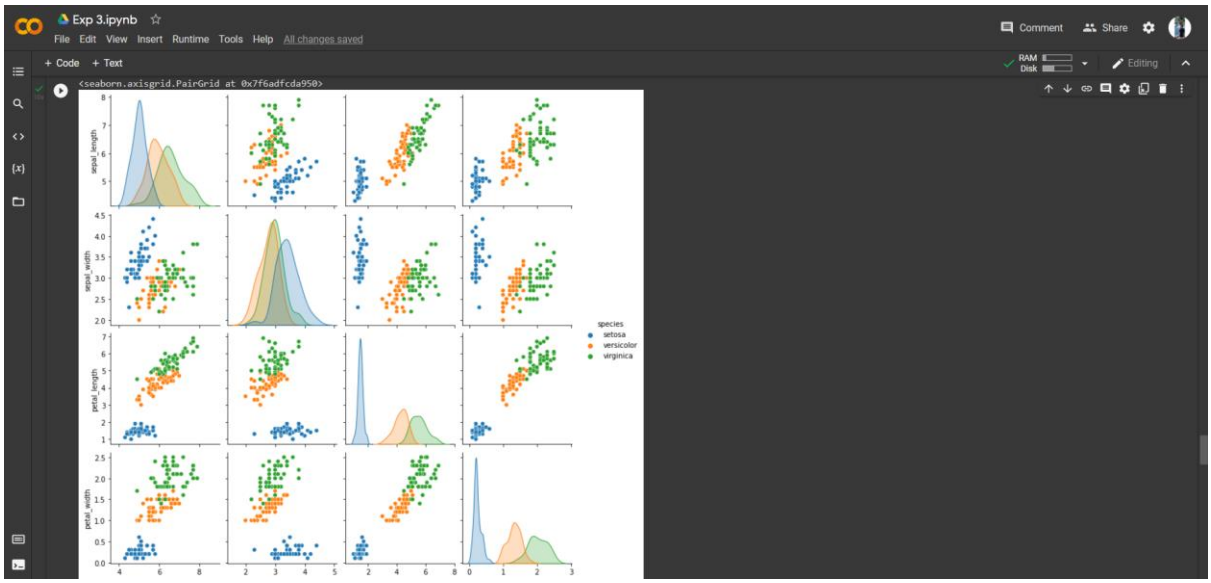
RAM Disk Editing

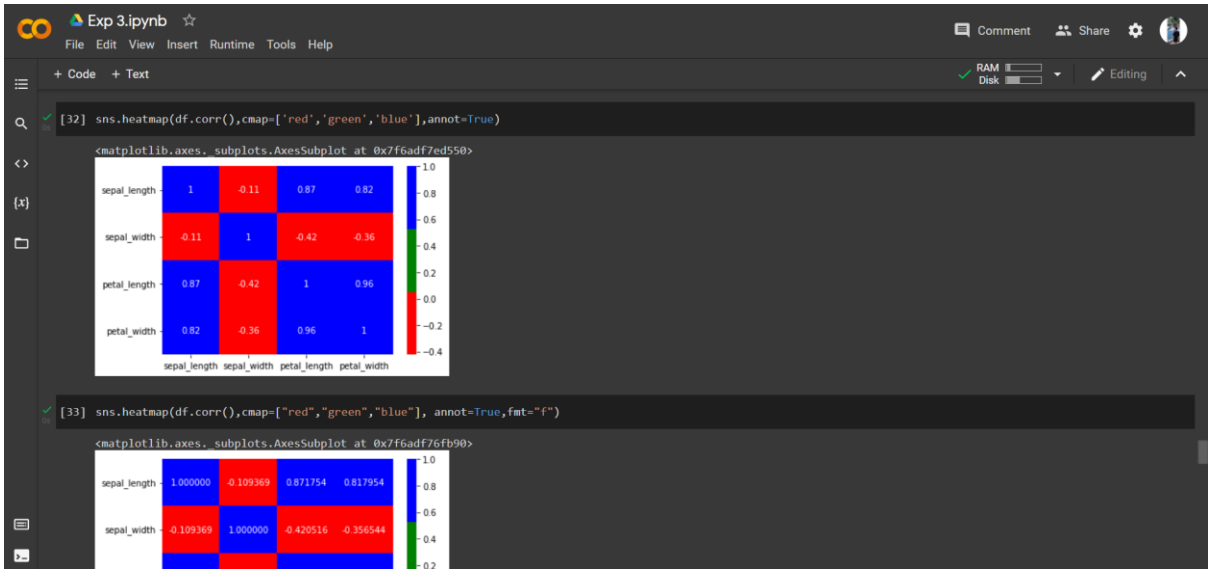
[28]




```
import seaborn as sns
import matplotlib.pyplot as plt

sns.pairplot(data=df, hue="species")
```


Exp 3.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
<seaborn.axisgrid.PairGrid at 0x7f6adfcda950>
sepal_length
sepal_width
petal_length
petal_width
species
setosa
versicolor
virginica


Exp 3.ipynb
File Edit View Insert Runtime Tools Help
Comment
Share
+ Code + Text
RAM
Disk
Editing
[32] sns.heatmap(df.corr(),cmap=['red','green','blue'],annot=True)
<matplotlib.axes._subplots.AxesSubplot at 0x7f6adf7ed550>
sepal_length 1 -0.11 0.87 0.82
sepal_width -0.11 1 -0.42 -0.36
petal_length 0.87 -0.42 1 0.96
petal_width 0.82 -0.36 0.96 1
sepal_length sepal_width petal_length petal_width
[33] sns.heatmap(df.corr(),cmap=["red","green","blue"], annot=True,fmt="f")
<matplotlib.axes._subplots.AxesSubplot at 0x7f6adf76fb90>
sepal_length 1.000000 -0.109369 0.871754 0.817954
sepal_width -0.109369 1.000000 -0.420516 -0.356544

Exp 3.ipynb

File Edit View Insert Runtime Tools Help All changes saved

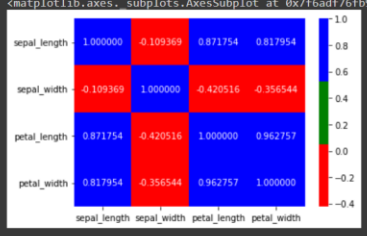
Comment Share

+ Code + Text

RAM Disk Editing

```
[33] sns.heatmap(df.corr(),cmap=["red","green","blue"], annot=True,fmt="f")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6adf76fb90>
```




```
[34] #normalization= rescaling values between [0,1]
     #standardization= mean=0,std.dev=1
```

```
[35] import pandas as pd
     df=pd.DataFrame({'y': [8, 12, 15, 14, 19, 23, 25, 29],
                      'x1': [5, 7, 7, 9, 12, 9, 9, 4],
                      'x2': [11, 8, 10, 6, 6, 5, 9, 12],
                      'x3': [2, 2, 3, 2, 5, 5, 7, 9]})
     df
     df_new = (df-df.mean())/df.std()
     print(df_new)
```

Exp 3.ipynb

File Edit View Insert Runtime Tools Help Saving...

Comment Share

+ Code + Text

RAM Disk Editing

```
[35]
          y        x1        x2        x3
0 -1.418032 -1.078639  1.025393 -0.908151
1 -0.857822 -0.294174 -0.146485 -0.908151
2 -0.437664 -0.294174  0.634767 -0.525772
3 -0.577717  0.490290 -0.927736 -0.908151
4  0.122546  1.666987 -0.927736  0.238987
5  0.682756  0.490290 -1.318362  0.238987
6  0.962861  0.490290  0.244141  1.003746
7  1.523071 -1.470871  1.416019  1.768505
```

```
[36] #define predictor variable columns
     df_x = df[['x1','x2','x3']]

     #standardize the values for each predictor variable
     df[['x1','x2','x3']]=(df_x-df_x.mean())/df_x.std()
     df
```

	y	x1	x2	x3
0	8	-1.078639	1.025393	-0.908151
1	12	-0.294174	-0.146485	-0.908151
2	15	-0.294174	0.634767	-0.525772
3	14	0.490290	-0.927736	-0.908151
4	19	1.666987	-0.927736	0.238987
5	23	0.490290	-1.318362	0.238987

Exp 3.ipynb

File Edit View Insert Runtime Tools Help

Comment Share

+ Code + Text

RAM Disk Editing

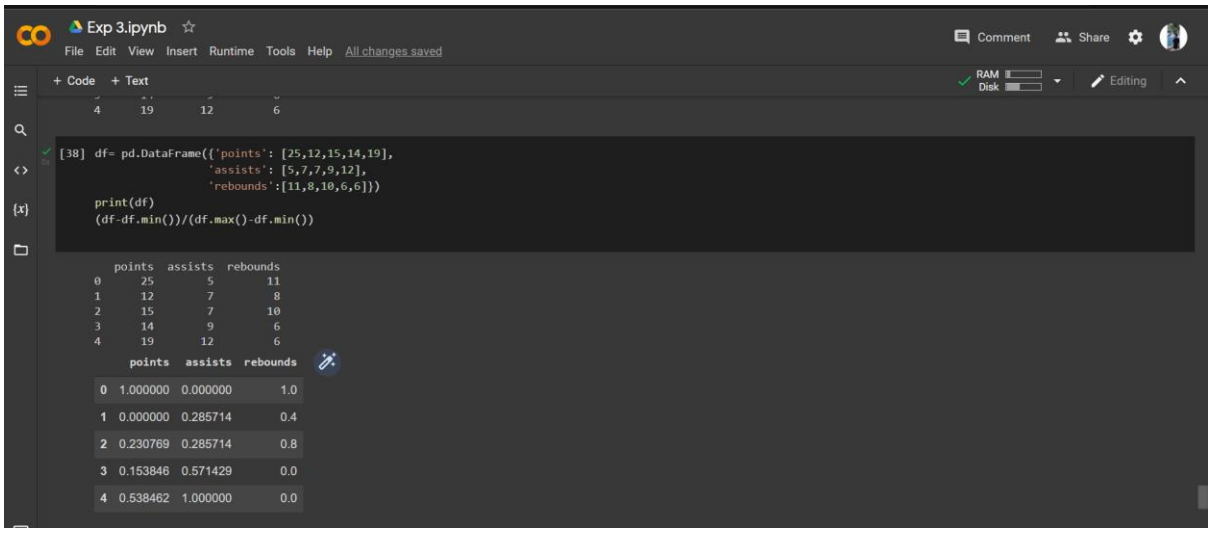
```
[37] import pandas as pd
     df= pd.DataFrame({'points': [25,12,15,14,19],
                       'assists': [5,7,7,9,12],
                       'rebounds':[11,8,10,6,6]})
     print(df)
```

```
   points  assists  rebounds
0      25        5        11
1      12        7         8
2      15        7        10
3      14        9         6
4      19       12         6
```

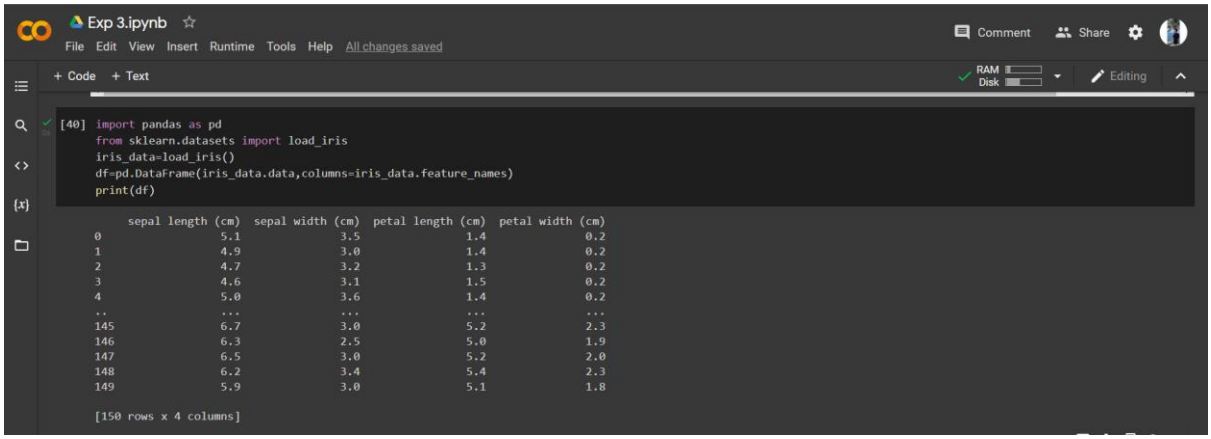
```
[38] df= pd.DataFrame({'points': [25,12,15,14,19],
                       'assists': [5,7,7,9,12],
                       'rebounds':[11,8,10,6,6]})
     print(df)
     (df-df.min())/(df.max()-df.min())
```

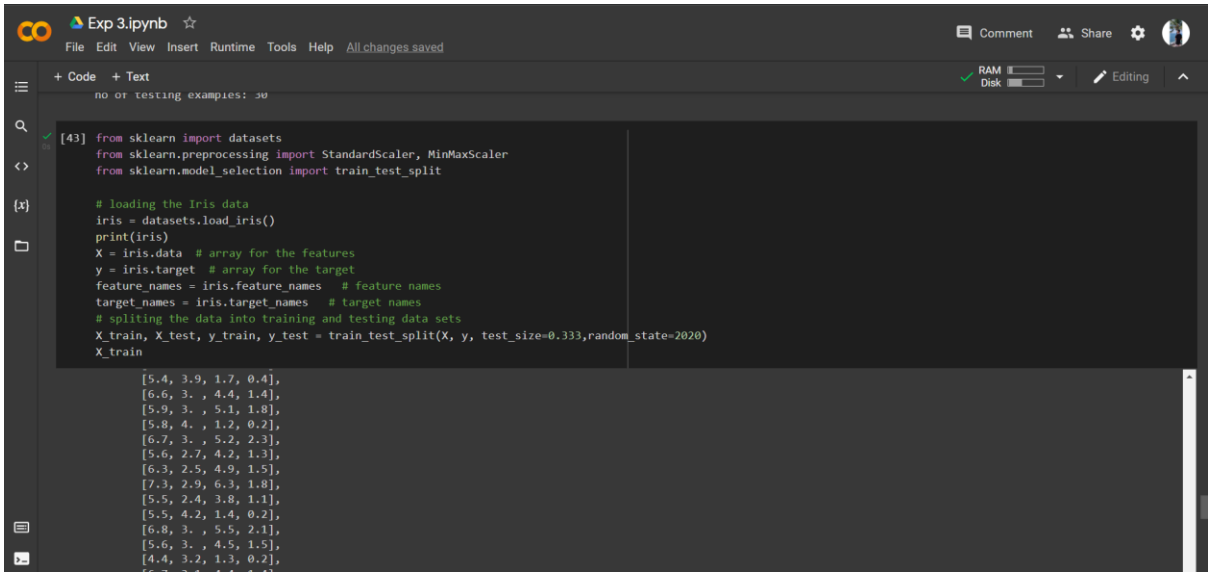
```
   points  assists  rebounds
0      25        5        11
1      12        7         8
2      15        7        10
3      14        9         6
4      19       12         6
```

	points	assists	rebounds
0	1.000000	0.000000	1.0


Exp 3.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
4 19 12 6
[38] df= pd.DataFrame({'points': [25,12,15,14,19],
'assists': [5,7,7,9,12],
'rebounds':[11,8,10,6,6]})
print(df)
(df-df.min())/(df.max()-df.min())
points assists rebounds
0 25 5 11
1 12 7 8
2 15 7 10
3 14 9 6
4 19 12 6
points assists rebounds
0 1.000000 0.000000 1.0
1 0.000000 0.285714 0.4
2 0.230769 0.285714 0.8
3 0.153846 0.571429 0.0
4 0.538462 1.000000 0.0


Exp 3.ipynb
File Edit View Insert Runtime Tools Help
Comment Share
+ Code + Text
RAM Disk
Editing
[39] import pandas as pd
df=pd.read_csv("https://raw.githubusercontent.com/uiuc-cse/data-fa14/gh-pages/data/iris.csv")
pd.crosstab(index=df['species'],columns=df['sepal_length'],margins=True)
sepal_length 4.3 4.4 4.5 4.6 4.7 4.8 4.9 5.0 5.1 5.2 5.3 5.4 5.5 5.6 5.7 5.8 5.9 6.0 6.1 6.2 6.3 6.4 6.5 6.6 6.7 6.8 6.9 7.0 7.1 7.2 7.3 7.
species
setosa 1 3 1 4 2 5 4 8 8 3 1 5 2 0 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
versicolor 0 0 0 0 0 0 1 2 1 1 0 1 5 5 5 3 2 4 4 2 3 2 1 2 3 1 1 1 0 0 0
virginica 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1 3 1 2 2 2 6 5 4 0 5 2 3 0 1 3 1
All 1 3 1 4 2 5 6 10 9 4 1 6 7 6 8 7 3 6 6 4 9 7 5 2 8 3 4 1 1 3 1
[40] import pandas as pd
from sklearn.datasets import load_iris
iris_data=load_iris()
df=pd.DataFrame(iris_data.data,columns=iris_data.feature_names)
print(df)

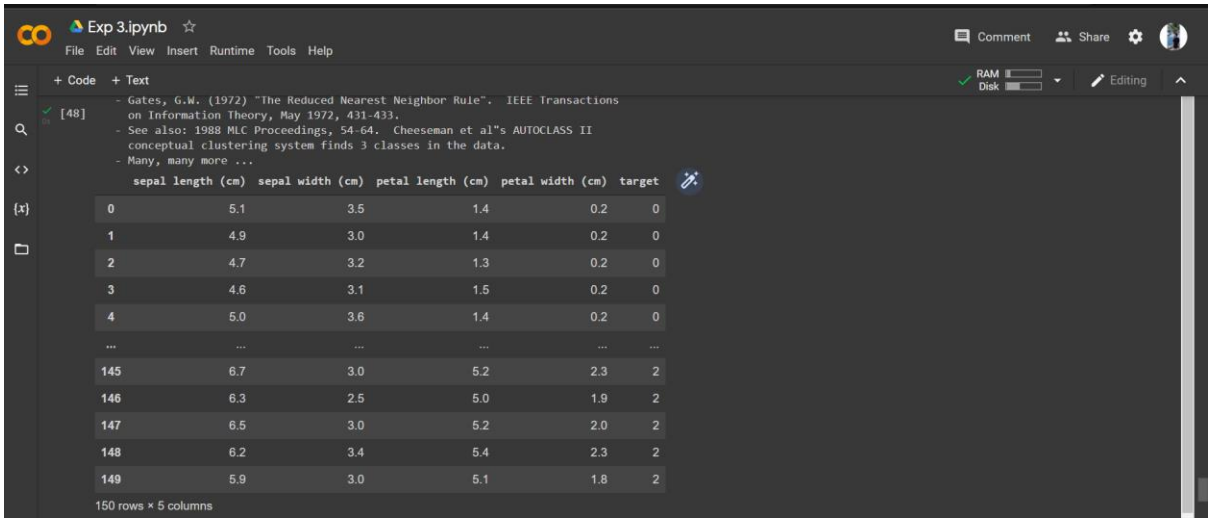

Exp 3.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[40] import pandas as pd
from sklearn.datasets import load_iris
iris_data=load_iris()
df=pd.DataFrame(iris_data.data,columns=iris_data.feature_names)
print(df)
sepal length (cm) sepal width (cm) petal length (cm) petal width (cm)
0 5.1 3.5 1.4 0.2
1 4.9 3.0 1.4 0.2
2 4.7 3.2 1.3 0.2
3 4.6 3.1 1.5 0.2
4 5.0 3.6 1.4 0.2
.. ..
145 6.7 3.0 5.2 2.3
146 6.3 2.5 5.0 1.9
147 6.5 3.0 5.2 2.0
148 6.2 3.4 5.4 2.3
149 5.9 3.0 5.1 1.8
[150 rows x 4 columns]

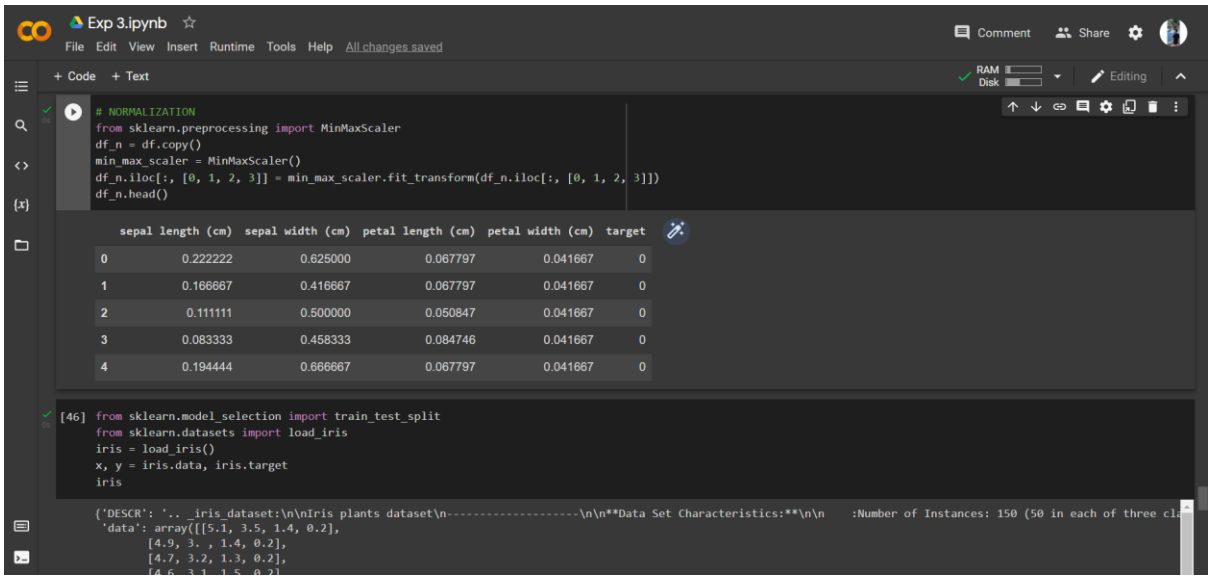


8. Post-Experiments Exercise

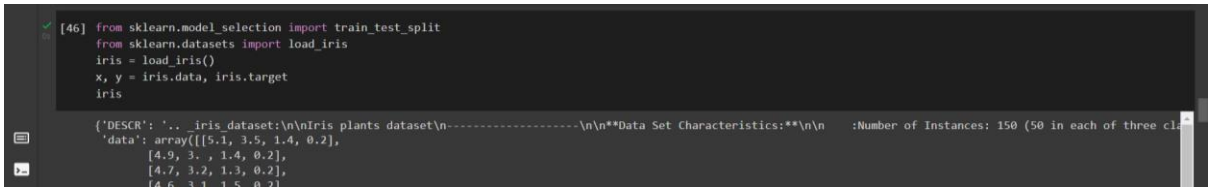
A. Extended Theory: (Soft Copy)

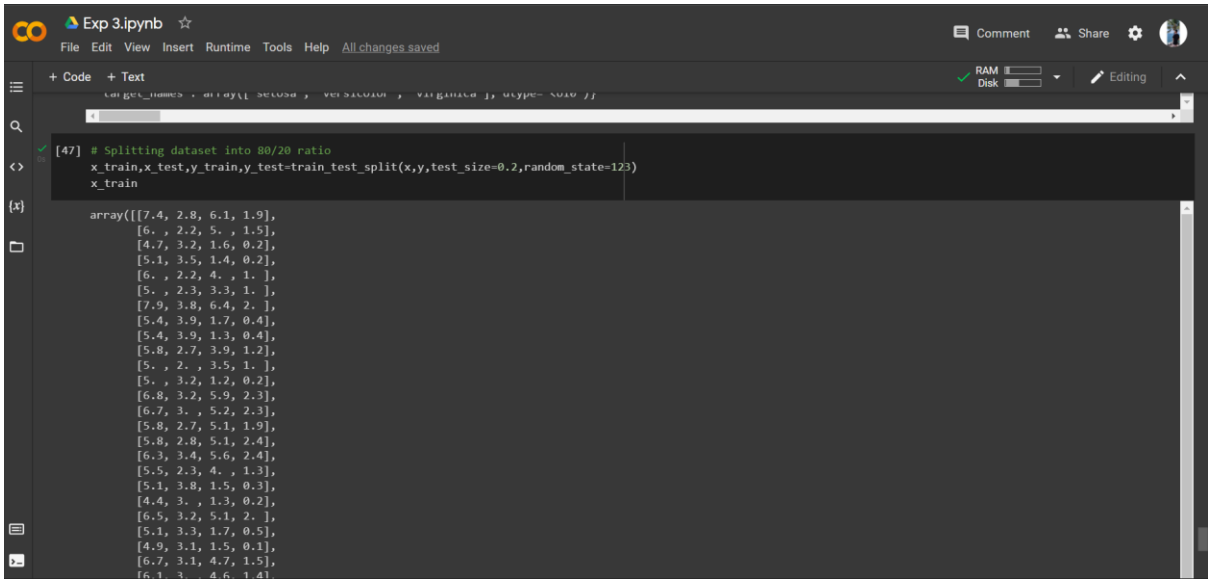
- Use iris dataset and perform rescaling using sklearn package using normalization





- Partition the iris dataset such that 80% data to be taken for training purpose





B. Questions:

- Mention three differences between normalization and standardization.
- Describe train_test_split function

Allan Rodrigues TE IT A-59 Rajdhani DATE / /

Q.8

B.

1) Mention three difference between normalization & standardization

Normalization	Standardization
1) Minimum & maximum value of features are used for scaling	1) Mean & standard deviation is used for scaling
2) It is used when features are of different scales	2) It is used when we want to ensure zero mean & unit standard deviation
3) Scales values between [0,1] or [-1, 1]	3) It is not bounded to a certain range

2) Describe train_test_split function.

~~test~~ train_test_split is a function in Sklearn model selection for splitting data arrays into two subsets for training data & for testing data. With this function, you don't need to ~~train~~ divide the dataset manually. By default, Sklearn train_test_split will make random partitions for the two subsets.

C. Conclusion:

Write the significance of the topic studied in the experiment.

Rajdhani DATE / /

C) Conclusion.

In this experiment we used the seaborn library for various plots like countplot, distribution plot and heatmap. We also learnt about normalization & standardization. We learnt to split the dataset into training and testing using train_test_split function from sklearn

D. References:

1. https://www.geeksforgeeks.org/exploratory-data-analysis-on-iris-dataset/
2. https://www.statology.org/normalize-columns-pandas-dataframe/Normalization
3. https://www.datascienceguide.org/python-code-snippets.html
4. https://machinelearningmastery.com/standardscaler-and-minmaxscaler-transforms-in-python/
