MACHINE LEARNING LAB

NAME:

Aaron Mathew

ROLL.NO:

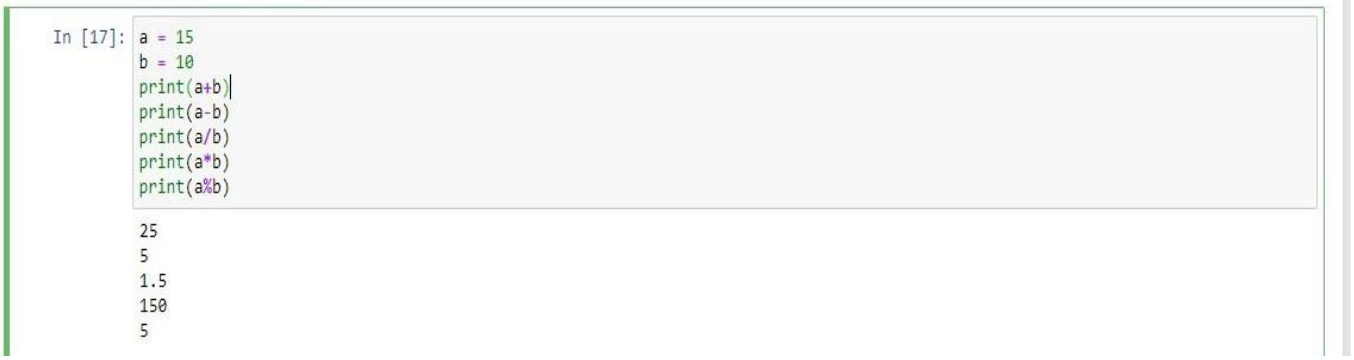
20BIS001

MLT-LAB-1 PYTHON BASICS

1. INSTALL JUPYTER NOTEBOOK AND TRY BASIC OPERATIONS:

Pandas in Python

Pandas is defined as an open-source library that provides high-performance data manipulation in Python. The name of Pandas is derived from the word Panel Data, which means an Econometrics from Multidimensional data. It is used for data analysis in Python anddeveloped by Wes McKinney in 2008.



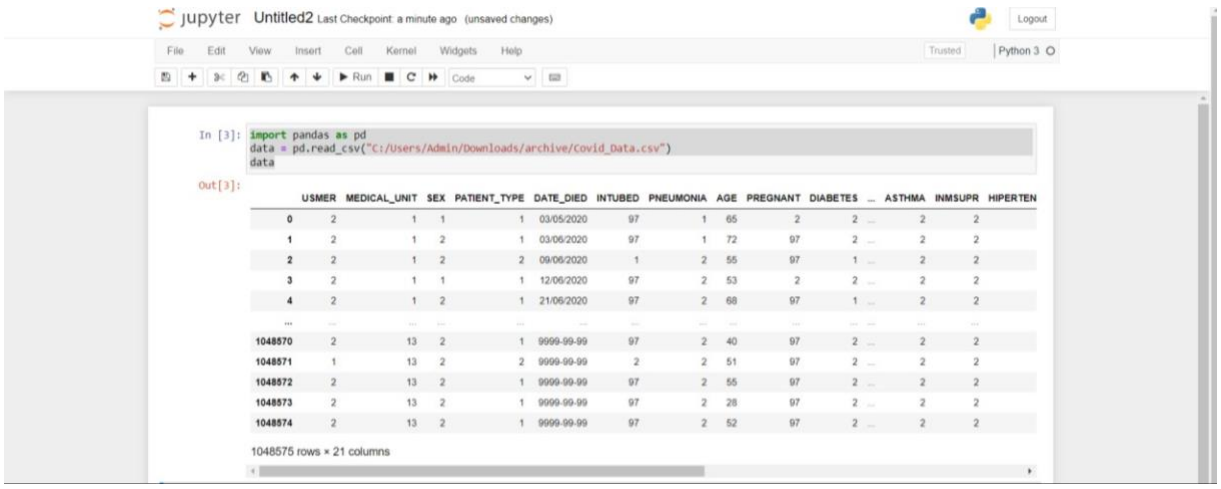
```
In [17]: a = 15
         b = 10
         print(a+b)
         print(a-b)
         print(a/b)
         print(a*b)
         print(a%b)

         25
         5
         1.5
         150
         5
```

2. DOWNLOAD A DATASET FROM KAGGLE AND UPLOAD USINGPYTHON:

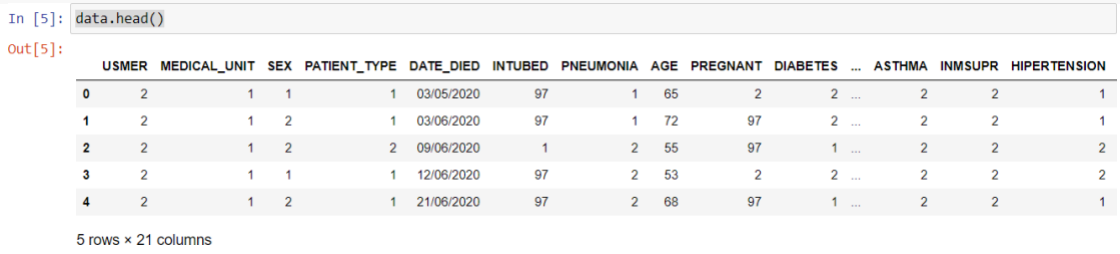
Reading CSV Data Using Pandas

Syntax: import pandas as pd



Syntax:

surveys_df.head() The head() method displays the first several lines o f a file. It



3. Display the column names:

TYPE FUNCTION:

```
In [6]: type(data)
Out[6]: pandas.core.frame.DataFrame
```

DataFrames have an attribute called dtypes

```
In [7]: data.dtypes
Out[7]: USMER                 int64
        MEDICAL_UNIT          int64
        SEX                   int64
        PATIENT_TYPE          int64
        DATE_DIED            object
        INTUBED               int64
        PNEUMONIA             int64
        AGE                   int64
        PREGNANT              int64
        DIABETES              int64
        COPD                  int64
        ASTHMA                int64
        INMSUPR               int64
        HIPERTENSION          int64
        OTHER_DISEASE         int64
        CARDIOVASCULAR        int64
        OBESITY               int64
        RENAL_CHRONIC         int64
        TOBACCO               int64
        CLASIFFICATION_FINAL  int64
        ICU                   int64
        dtype: object
```

4. CREATE A NEW COLUMN

Groups in Pandas

We can calculate basic statistics for all records in a single column using the syntax below

```
In [10]: data['INTUBED'].describe()
Out[10]: count    1.048575e+06
         mean     7.952288e+01
         std      3.686889e+01
         min      1.000000e+00
         25%      9.700000e+01
         50%      9.700000e+01
         75%      9.700000e+01
         max      9.900000e+01
         Name: INTUBED, dtype: float64
```

```
In [12]: data['INTUBED'].unique()
Out[12]: array([97,  1,  2, 99], dtype=int64)
```

```
In [13]: data['INTUBED'].count()
Out[13]: 1048575
```

5. Using group by statistics can be understood in an easier method.

```
In [14]: grouped_by_age = data.groupby('AGE')
In [15]: grouped_by_age.describe()
Out[15]:
```

	USMER								MEDICAL_UNIT		...	CLASIFFICA
	count	mean	std	min	25%	50%	75%	max	count	mean	...	75%
AGE												
0	3862.0	1.539358	0.498513	1.0	1.0	2.0	2.0	2.0	3862.0	9.283791	...	7.00
1	4802.0	1.445648	0.497089	1.0	1.0	1.0	2.0	2.0	4802.0	9.338192	...	7.00
2	3178.0	1.480176	0.499685	1.0	1.0	1.0	2.0	2.0	3178.0	9.113908	...	7.00
3	2559.0	1.509965	0.499998	1.0	1.0	2.0	2.0	2.0	2559.0	9.141462	...	7.00
4	2485.0	1.517907	0.499780	1.0	1.0	2.0	2.0	2.0	2485.0	9.354527	...	7.00
...	...	...	...	...	...	...	...	...	...	...	...	...
117	3.0	1.333333	0.577350	1.0	1.0	1.0	1.5	2.0	3.0	12.000000	...	6.00
118	2.0	2.000000	0.000000	2.0	2.0	2.0	2.0	2.0	2.0	12.000000	...	6.75
119	3.0	1.666667	0.577350	1.0	1.5	2.0	2.0	2.0	3.0	10.000000	...	7.00
120	5.0	2.000000	0.000000	2.0	2.0	2.0	2.0	2.0	5.0	11.400000	...	7.00
121	1.0	2.000000	NaN	2.0	2.0	2.0	2.0	2.0	1.0	12.000000	...	6.00

121 rows × 152 columns

```
In [16]: grouped_by_age.mean()
```

Out[16]:

AGE	USMER	MEDICAL_UNIT	SEX	PATIENT_TYPE	INTUBED	PNEUMONIA	PREGNANT
0	1.539358	9.283791	1.546867	1.645520	36.491973	2.835060	54.149922
1	1.445648	9.338192	1.566639	1.446064	63.340691	14.086006	56.749896
2	1.480176	9.113908	1.560101	1.329138	72.909692	15.476400	56.236627
3	1.509965	9.141462	1.528331	1.255569	77.825713	14.176241	52.716295
4	1.517907	9.354527	1.524346	1.261167	76.158149	14.040241	52.469618
...	...	...	...	...	...	...	...
117	1.333333	12.000000	1.333333	1.000000	97.000000	2.000000	33.666667
118	2.000000	12.000000	2.000000	1.500000	49.500000	1.500000	97.000000
119	1.666667	10.000000	1.000000	1.000000	97.000000	1.666667	2.000000
120	2.000000	11.400000	1.800000	1.000000	97.000000	1.800000	97.200000
121	2.000000	12.000000	1.000000	1.000000	97.000000	2.000000	2.000000

121 rows × 19 columns

6. MANIPULATE TWO COLUMNS AND POPULATE THE NEWLY CREATE DONE:

Count the number of samples by species.

```
In [19]: age_count = data.groupby('AGE')['MEDICAL_UNIT'].count()

In [20]: print(age_count)

AGE
0      3862
1      4802
2      3178
3      2559
4      2485
       ...
117       3
118       2
119       3
120       5
121       1
Name: MEDICAL_UNIT, Length: 121, dtype: int64

In [ ]:
```

7. SELECT FEW ROWS ON SOME CONDITIONS AND APPLY

Calculating Statistics From Data In A Pandas DataFrame

Look at the column names

```
In [21]: data.columns
Out[21]: Index(['USMER', 'MEDICAL_UNIT', 'SEX', 'PATIENT_TYPE', 'DATE_DIED', 'INTUBED',
       'PNEUMONIA', 'AGE', 'PREGNANT', 'DIABETES', 'COPD', 'ASTHMA', 'INMSUPR',
       'HIPERTENSION', 'OTHER_DISEASE', 'CARDIOVASCULAR', 'OBESITY',
       'RENAL_CHRONIC', 'TOBACCO', 'CLASIFFICATION_FINAL', 'ICU'],
      dtype='object')
```

The pd.unique function tells us all of the unique values in the column.

8. SELECT FEW COLUMNS ON SOME CONDITIONS AND APPLY

TO DISPLAY A PARTICULAR COLOUM

SYNTAX: data['AGE']

```
In [22]: data['AGE']
Out[22]: 0          65
         1          72
         2          55
         3          53
         4          68
                    ..
         1048570    40
         1048571    51
         1048572    55
         1048573    28
         1048574    52
         Name: AGE, Length: 1048575, dtype: int64
```

Plotting Data Using Pandas

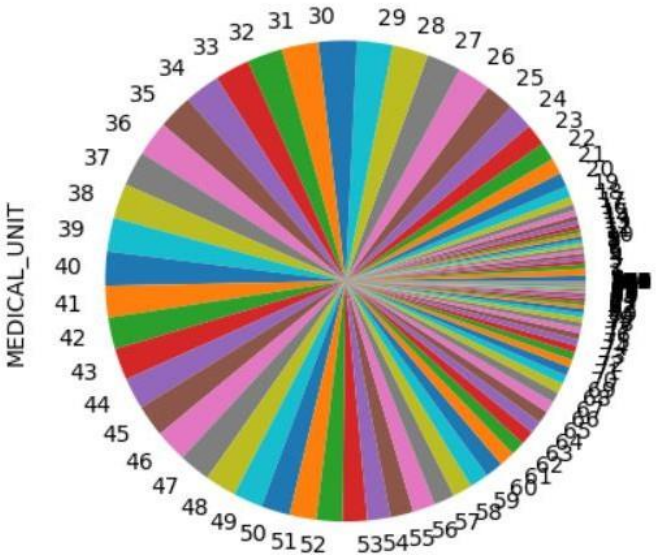
We can plot our summary stats using Pandas, too.

Make sure figures appear inline in python Notebook %matplotlib inline

Create a quick bar chart species_counts.plot(kind='pie');

```
In [23]: %matplotlib inline

In [24]: age_count.plot(kind='pie');
```



MATPLOTLIB:

SYNTAX: import matplotlib print(matplotlib._version__)

```
In [25]: import matplotlib

In [26]: print(matplotlib.__version__)
         3.5.2
```

9. ARRAY OF NUMBERS:

import pandas as pd

df=pd.DataFrame({'d1':[10,20,30,40,50],'d2':[30,50,60,20,10]}) df

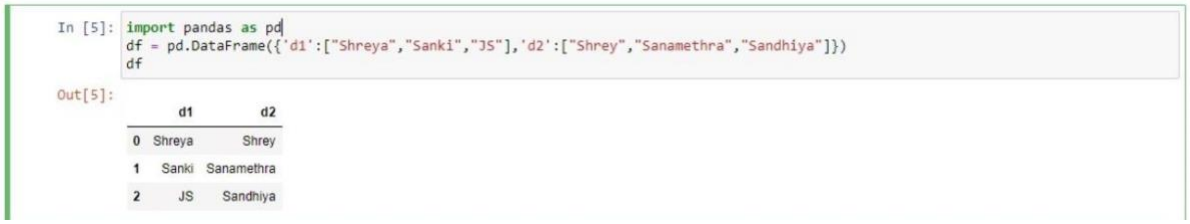
```
In [4]: import pandas as pd
        df = pd.DataFrame({'d1':[10,20,30,40,50],'d2':[30,40,50,60,70]})
        df

Out[4]:
```

	d1	d2
0	10	30
1	20	40
2	30	50
3	40	60
4	50	70

10. ARRAY OF STRING:

df1=pd.DataFrame({'d1':["ML","IWP"],'d2':["Good","Good"]}) df1



```
In [5]: import pandas as pd
        df = pd.DataFrame({'d1':["Shreya","Sanki","JS"],'d2':["Shrey","Sanamethra","Sandhiya"]})
        df
```

Out[5]:

	d1	d2
0	Shreya	Shrey
1	Sanki	Sanamethra
2	JS	Sandhiya

11. MATRIX IN NUMPY

```
In [16]: import numpy as np
         mat = np.array([[10,20,30],[30,40,50],[60,70,80]])
         mat

Out[16]: array([[10, 20, 30],
                [30, 40, 50],
                [60, 70, 80]])
```

12. CORRELATION MATRIX:

cor=df.corr(

)Cor

```
In [10]: import pandas as pd
         df = pd.DataFrame({'d1':[10,20,30,40,50],'d2':[30,40,50,60,70]})
         df.corr()
```

Out[10]:

	d1	d2
d1	1.0	1.0
d2	1.0	1.0