Unit - 2

1) Data Aggeregations:

* Aggeregation is a process of implementing any Mathematical operation on a dataset.

* Aggeregation is a one of techinque used in pandas to manipulate the data in dataframe.

* Dataframe.aggregation() function is used to apply aggregation.

Sum: It returns the Sum of the values

Min: Returs the minimum of value.

max: Returns maximum of value.

* Since aggregation only works with numeric type columns.

* Let us consider a numeric Column dataset.

	length	width	hargn	price
0		0.89027	48.8	13450
1	0.8118			
2	0.8114	0.82709	48.8	16500
3	0.8124	0.909722	52.4	16500
4	0.824	0.901944	54.3	13500.
:				
:				
.				

* Now let's apply aggergation function to the table by

#applying aggergation

dataset.agg("mean", axis="rows").

The output.

length 0.837102

width 0.915126.

height: 53.7667.

price: 13207

2) Data manipulation using Pandas:

*Pandas is a powerful data manipulation library for Python.

* It provides data structures like Dataframes and series.

* It is also contains variety functions to manipulate and analyze data.

* Here is a breif overview of some common data manipulation tasks using pandas.

1. Reading and writing Data
2. Exploring Data
3. Selecting Data
4. Cleaning Data
5. Manipulating columns
6. Grouping and Aggregating
7. Merging Dataframes
8. Time Series Operations.

*Reading and writing Data:

```
import pandas as pd.
df = pd.read_csv('your data.csv')
df.to_csv('output.csv', index=False)
```

* Exploring Data:

```
df.head() // displays first five rows of
            Dataframe.
df.describe() // displays basic statistics
                of Dataframe
df.info() // Display information
             about dataframe
```

* Selecting Data:

```
Column = df['Column_name']
// Selecting a single Column
Colums = df[['Colum1_','Columna']]
// Selecting a multiple Columns.
```

Cleaning Data:

df.dropna // Drops rows with missing values.

df.fillna(value) // fill missing values with a specific value.

df.drop_duplicates() // removes duplicate value.

df.replace(old_value, new_value).

Manipulating columns.

#Adding new columns.

df['new_colum'] = df['colum1'] + df['colum

#drop columns.

df.drop['column1', 'columng']

Grouping and aggeration.

df.groupby('columnname')

// aggregation .agg('sum', 'mean', 'count').

Merging Dataframes;

```
result = pd.concat([df1, df2])
// concatenation.

//Merging
pd.merge(df1, df2).
```

3. Pandas Objects:

* In pandas, the primary data structures are Dataframe and Series.

* which are built on top of numpy.

* These structures are designed to handle data manipulations

Dataframe:

* A two dimensional labeled data structure with columns that can be of different types (numeric, string boolean, etc...)

import pandas as pd.

```
data = {'column1': [1, 2, 3], 'column2': ['A', 'B', 'C']}
df = pd.DataFrame(data)
```

Attributes of dataframe:

df.shape: Returns the number of rows and columns.

df.columns: Returns column labels

df.index: Returns row labels.

Series:

A one-dimensional labeled array capable of holding any data type.

```
import pandas as pd.

data = [1, 2, 3, 4]

Series = pd.Series(data).
```

attribute:

- Series.values : Returns the data as a Numpy array.

Series.index : Returns the index labels

4) Indexing:

* In pandas, data indexing refers to the process of selecting and retrieving specific subsets of data from dataframes or series.

* Indexing in pandas can be done using flexiblity for accessing rows, columns or individual elements

* selection by Label:

- loc[]
- uses labels to select data.
- Allows row selecting and columns by label

* Selection by position:

- .iloc[]:

* Uses integer position to select data.

* Allows row selecting and column selecting.

* Conditional indexing:

* Uses conditions to select data.

df['Age'] > 30]

* Multiindexing

* It is also known as hierachical indexing.

* It allows you to have multiple index levels on an axis.

5) ✱ Combining datasets:

- ✱ Merge
- ✱ concat
- ✱ apperd
- ✱ Join

6. Vectorized String operations:

1) Accessing Elements.

```
import pandas as pd

Str = "sherbin"

printf (Str.get (0)).
```

output

S.

2. changing case.

```
import pandas as pd

Str = "sherbin"

Str2 = "SHERBIN"

Str2.lower()

Str. upper()
```

Output: Sherbin, SHERBIN

3. String length:

```
import pandas as pd.

Str = "Sherbin"

Str. len()
```

output

7

4. String Concatenation:

```
Pmport pandas. as Pd.

Str = "Hello"

Str2 = "world"

Str3 = Str + Str2
```

output

Hello world.

5. Substring Matching:

```
Str = "working"

Str. contain (or):
```

Output:

True.

* String replacement:

```
Str = "Sherbin";
str.replace("b", "l");
```

output

Sherlin.