EDA- Data Farames

A **DataFrame** is a 2-dimensional labeled data structure.

- Think of it as a table with rows and columns.
- It consists of one or more **Series** (columns).
- Often referred to as df in Python.
- Commonly used for data manipulation and analysis

Loading a DataFrame:

```
df = pd.read_csv('file.csv')
```

View the first few (by default 5) rows

```
df.head()
```

View the last few (by default 5) rows

```
df.tail()
```

- **What are NaN values?**
- Stands for **Not a Number**.
- It comes from **numpy.nan**, which is a special floating-point value
- representing "missing" or "undefined" data.
- **NaN** is always treated as a floating-point value, even in columns of mixed types.
- Operations with **NaN** often result in **NaN**

  (e.g., 1 + np.nan = np.nan), preserving the missing data state.

DataFrame Attributes

- **df.index**: The row labels or index of the DataFrame.
- **df.values**: The data values in the DataFrame as a NumPy array.
- **df.shape**: The dimensions of the DataFrame (rows, columns).
- **df.dtypes**: Data types of each column.
- **df.axes**: A list of row and column labels.
- **df.columns**: The column labels of the DataFrame.

DataFrame Methods

df.info()

Summary of the DataFrame, including data types and non-null counts.

```
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   sepal length (cm)  150 non-null    float64
 1   sepal width (cm)   150 non-null    float64
 2   petal length (cm)  150 non-null    float64
 3   petal width (cm)   150 non-null    float64
 4   target             150 non-null    int64  
dtypes: float64(4), int64(1)
memory usage: 6.0 KB
```

DataFrame Methods

df.describe()

Statistical summary of numerical columns (e.g., mean, std, min, max).

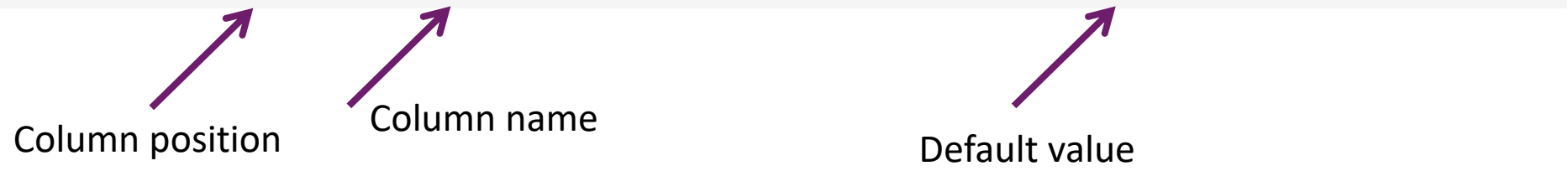
	sepal length (cm)	sepal width (cm)	petal length (cm)	\
count	150.000000	150.000000	150.000000	
mean	5.843333	3.057333	3.758000	
std	0.828066	0.435866	1.765298	
min	4.300000	2.000000	1.000000	
25%	5.100000	2.800000	1.600000	
50%	5.800000	3.000000	4.350000	
75%	6.400000	3.3000	5.100000	
max	7.900000	4.4000	6.900000	

DataFrame Methods

to create a new column:

```
df['new_column']= 'default value'
```

```
df.insert(2 ,column= 'column name',value = 'default')
```



DataFrame Methods for broadcasting

DataFrame.add : Add DataFrames.
DataFrame.sub : Subtract DataFrames.
DataFrame.mul : Multiply DataFrames.
DataFrame.div : Divide DataFrames (float division).
DataFrame.truediv : Divide DataFrames (float division).
DataFrame.floordiv : Divide
DataFrames (integer division).
DataFrame.mod : Calculate modulo (remainder after division).
DataFrame.pow : Calculate exponential power.

DataFrame Methods

df[‘column name’].count()- how many values

df[‘column name’].value_counts – counts how many values in each “category”

df[‘column name’].nunique() is like df[‘column name’].value_counts.count()

How many “categories” are there.

df[‘column name’].unidue()- returns an array of all “categories”

DataFrame Methods for statistice

df[‘column name’].mode()

df[‘column name’].mean()

df[‘column name’].std()

df[‘column name’].var()

df[‘column name’].mode()

df.quantile(*Value between 0 <= q <= 1*)