🔹 Additional Pandas Series Methods

✅ astype() – Type Conversion

Used to change the data type of the Series elements.

```
s = pd.Series([1, 2, 3])

s = s.astype(float)      # Convert int to float

s = s.astype(str)        # Convert to string
```

👉 Useful when cleaning data (e.g., converting strings to numbers or vice versa).

✅ between() – Range Filter

Checks if values lie between two values (inclusive by default).

```
s = pd.Series([10, 20, 30, 40])

s.between(15, 35)

# Output: [False, True, True, False]


# Can also be used with boolean indexing:

s[s.between(15, 35)]
```

✅ clip() – Limit Values

Restricts values to a specified range.

```
s = pd.Series([5, 10, 15, 20])

s.clip(lower=8, upper=18)

# Output: [8, 10, 15, 18]
```

👉 Very useful in outlier treatment or capping.

✅ drop_duplicates() – Remove Duplicate Values

Removes duplicate values in the Series.

s = pd.Series([1, 2, 2, 3, 3, 3])

s.drop_duplicates()

Output: [1, 2, 3]

Optional:

s.drop_duplicates(keep='last') # Keep last occurrence

s.drop_duplicates(keep=False)   # Drop all duplicates

✅ duplicated() – Detect Duplicates

Returns a Boolean Series indicating duplicate status.

s = pd.Series([1, 2, 2, 3, 3, 3])

s.duplicated()

Output: [False, False, True, False, True, True]

Use with filtering:

s[s.duplicated()]

✅ isnull() – Detect Missing Values

Returns True for NaN/missing values.

s = pd.Series([1, None, 3, np.nan])

s.isnull()

Output: [False, True, False, True]

✅ dropna() – Remove Missing Values

Removes NaN values from the Series.

s = pd.Series([1, None, 3])

s.dropna()

Output: [1, 3]

s.dropna(inplace=True) # Modify in place

✅ fillna() – Fill Missing Values

Replaces NaN with a specified value or method.

s = pd.Series([1, None, 3])

s.fillna(0)

Output: [1, 0, 3]

Other options:

s.fillna(method='ffill')   # Forward fill

s.fillna(method='bfill')   # Backward fill

✅ isin() – Check Membership

Returns True for elements present in a given list.

s = pd.Series(['a', 'b', 'c', 'a'])

s.isin(['a', 'c'])

Output: [True, False, True, True]

Filtering

s[s.isin(['a', 'c'])]

✅ apply() – Apply a Function to Each Element

Applies a custom or built-in function element-wise.

s = pd.Series([1, 2, 3])

s.apply(lambda x: x ** 2)

Output: [1, 4, 9]

Also works with defined functions:

def convert(x):

```
    return x + 5


s.apply(convert)
```

✅ copy() – Create a Deep Copy

Creates an independent copy of the Series.

```
s1 = pd.Series([1, 2, 3])

s2 = s1.copy()


s2[0] = 100

# s1 remains unchanged
```

👉 Useful to avoid modifying the original data accidentally.
