



# Working with Missing Data in Pandas

[Read](#)

[Discuss](#)

[Courses](#)

[Practice](#)

Missing Data can occur when no information is provided for one or more items or for a whole unit. Missing Data is a very big problem in a real-life scenarios. Missing Data can also refer to as NA(Not Available) values in pandas. In DataFrame sometimes many datasets simply arrive with missing data, either because it exists and was not collected or it never existed. For Example, Suppose different users being surveyed may choose not to share their income, some users may choose not to share the address in this way many datasets went missing.



In Pandas missing data is represented by two value:

- **None:** None is a Python singleton object that is often used for missing data in Python code.
- **NaN :** NaN (an acronym for Not a Number), is a special floating-point value recognized by all systems that use the standard IEEE floating-point representation



AD

Pandas treat None and NaN as essentially interchangeable for indicating missing or null values. To facilitate this convention, there are several useful functions for detecting, removing, and replacing null values in Pandas DataFrame :

- [isnull\(\)](#).
- [notnull\(\)](#).
- [dropna\(\)](#).
- [fillna\(\)](#).
- [replace\(\)](#).
- [interpolate\(\)](#).

In this article we are using CSV file, to download the CSV file used, Click [Here](#).

### Checking for missing values using isnull() and notnull()

In order to check missing values in Pandas DataFrame, we use a function isnull() and notnull(). Both function help in checking whether a value is NaN or not. These function can also be used in Pandas Series in order to find null values in a series.

#### Checking for missing values using isnull()

In order to check null values in Pandas DataFrame, we use isnull() function this function return dataframe of Boolean values which are True for NaN values.

**Code #1:**

---

**Python**

```
# importing pandas as pd
import pandas as pd

# importing numpy as np
import numpy as np

# dictionary of lists
dict = {'First Score':[100, 90, np.nan, 95],
        'Second Score': [30, 45, 56, np.nan],
        'Third Score':[np.nan, 40, 80, 98]}

# creating a dataframe from list
df = pd.DataFrame(dict)

# using isnull() function
df.isnull()
```

|   | First Score | Second Score | Third Score |
|---|-------------|--------------|-------------|
| 0 | False       | False        | True        |
| 1 | False       | False        | False       |
| 2 | True        | False        | False       |
| 3 | False       | True         | False       |

Output:

Code #2:

---

## Python

```
# importing pandas package
import pandas as pd

# making data frame from csv file
data = pd.read_csv("employees.csv")

# creating bool series True for NaN values
bool_series = pd.isnull(data["Gender"])

# filtering data
# displaying data only with Gender = NaN
data[bool_series]
```

**Output:** As shown in the output image, only the rows having Gender = NULL are displayed.

|     | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team                 |
|-----|------------|--------|------------|-----------------|--------|---------|-------------------|----------------------|
| 20  | Lois       | NaN    | 4/22/1995  | 7:18 PM         | 64714  | 4.934   | True              | Legal                |
| 22  | Joshua     | NaN    | 3/8/2012   | 1:58 AM         | 90816  | 18.816  | True              | Client Services      |
| 27  | Scott      | NaN    | 7/11/1991  | 6:58 PM         | 122367 | 5.218   | False             | Legal                |
| 31  | Joyce      | NaN    | 2/20/2005  | 2:40 PM         | 88657  | 12.752  | False             | Product              |
| 41  | Christine  | NaN    | 6/28/2015  | 1:08 AM         | 66582  | 11.308  | True              | Business Development |
| 49  | Chris      | NaN    | 1/24/1980  | 12:13 PM        | 113590 | 3.055   | False             | Sales                |
| 51  | NaN        | NaN    | 12/17/2011 | 8:29 AM         | 41126  | 14.009  | NaN               | Sales                |
| 53  | Alan       | NaN    | 3/3/2014   | 1:28 PM         | 40341  | 17.578  | True              | Finance              |
| 60  | Paula      | NaN    | 11/23/2005 | 2:01 PM         | 48866  | 4.271   | False             | Distribution         |
| 64  | Kathleen   | NaN    | 4/11/1990  | 6:46 PM         | 77834  | 18.771  | False             | Business Development |
| 69  | Irene      | NaN    | 7/14/2015  | 4:31 PM         | 100863 | 4.382   | True              | Finance              |
| 70  | Todd       | NaN    | 6/10/2003  | 2:26 PM         | 84692  | 6.617   | False             | Client Services      |
| ⋮   | ⋮          | ⋮      | ⋮          | ⋮               | ⋮      | ⋮       | ⋮                 | ⋮                    |
| 939 | Ralph      | NaN    | 7/28/1995  | 6:53 PM         | 70635  | 2.147   | False             | Client Services      |
| 945 | Gerald     | NaN    | 4/15/1989  | 12:44 PM        | 93712  | 17.426  | True              | Distribution         |
| 961 | Antonio    | NaN    | 6/18/1989  | 9:37 PM         | 103050 | 3.050   | False             | Legal                |
| 972 | Victor     | NaN    | 7/28/2006  | 2:49 PM         | 76381  | 11.159  | True              | Sales                |
| 985 | Stephen    | NaN    | 7/10/1983  | 8:10 PM         | 85668  | 1.909   | False             | Legal                |
| 989 | Justin     | NaN    | 2/10/1991  | 4:58 PM         | 38344  | 3.794   | False             | Legal                |
| 995 | Henry      | NaN    | 11/23/2014 | 6:09 AM         | 132483 | 16.655  | False             | Distribution         |

145 rows x 8 columns

### Checking for missing values using notnull()

In order to check null values in Pandas Dataframe, we use notnull() function this function return dataframe of Boolean values which are False for NaN values.

#### Code #3:

## Python

```
# importing pandas as pd
import pandas as pd

# importing numpy as np
import numpy as np

# dictionary of lists
dict = {'First Score':[100, 90, np.nan, 95],
        'Second Score': [30, 45, 56, np.nan],
        'Third Score':[np.nan, 40, 80, 98]}

# creating a dataframe using dictionary
df = pd.DataFrame(dict)

# using notnull() function
df.notnull()
```

#### Output:

|   | First Score | Second Score | Third Score |
|---|-------------|--------------|-------------|
| 0 | True        | True         | False       |
| 1 | True        | True         | True        |
| 2 | False       | True         | True        |
| 3 | True        | False        | True        |

Code #4:

---

## Python

```
# importing pandas package
import pandas as pd

# making data frame from csv file
data = pd.read_csv("employees.csv")

# creating bool series True for NaN values
bool_series = pd.notnull(data["Gender"])

# filtering data
# displaying data only with Gender = Not NaN
data[bool_series]
```

**Output:** As shown in the output image, only the rows having Gender = NOT NULL are displayed.

|     | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team                 |
|-----|------------|--------|------------|-----------------|--------|---------|-------------------|----------------------|
| 0   | Douglas    | Male   | 8/6/1993   | 12:42 PM        | 97308  | 6.945   | True              | Marketing            |
| 1   | Thomas     | Male   | 3/31/1996  | 6:53 AM         | 61933  | 4.170   | True              | NaN                  |
| 2   | Maria      | Female | 4/23/1993  | 11:17 AM        | 130590 | 11.858  | False             | Finance              |
| 3   | Jerry      | Male   | 3/4/2005   | 1:00 PM         | 138705 | 9.340   | True              | Finance              |
| 4   | Larry      | Male   | 1/24/1998  | 4:47 PM         | 101004 | 1.389   | True              | Client Services      |
| 5   | Dennis     | Male   | 4/18/1987  | 1:35 AM         | 115163 | 10.125  | False             | Legal                |
| 6   | Ruby       | Female | 8/17/1987  | 4:20 PM         | 65476  | 10.012  | True              | Product              |
| 7   | NaN        | Female | 7/20/2015  | 10:43 AM        | 45906  | 11.598  | NaN               | Finance              |
| 8   | Angela     | Female | 11/22/2005 | 6:29 AM         | 95570  | 18.523  | True              | Engineering          |
| 9   | Frances    | Female | 8/8/2002   | 6:51 AM         | 139852 | 7.524   | True              | Business Development |
| ⋮   | ⋮          | ⋮      | ⋮          | ⋮               | ⋮      | ⋮       | ⋮                 | ⋮                    |
| 994 | George     | Male   | 6/21/2013  | 5:47 PM         | 98874  | 4.479   | True              | Marketing            |
| 996 | Phillip    | Male   | 1/31/1984  | 6:30 AM         | 42392  | 19.675  | False             | Finance              |
| 997 | Russell    | Male   | 5/20/2013  | 12:39 PM        | 96914  | 1.421   | False             | Product              |
| 998 | Larry      | Male   | 4/20/2013  | 4:45 PM         | 60500  | 11.985  | False             | Business Development |
| 999 | Albert     | Male   | 5/15/2012  | 6:24 PM         | 129949 | 10.169  | True              | Sales                |

855 rows x 8 columns

## Filling missing values using fillna(), replace() and interpolate()

In order to fill null values in a datasets, we use fillna(), replace() and interpolate() function these function replace NaN values with some value of their own. All these function help in filling a null values in datasets of a DataFrame. Interpolate() function is basically used to fill NA values in the dataframe but it uses various interpolation technique to fill the missing values rather than hard-coding the value.

**Code #1:** Filling null values with a single value

## Python

```
# importing pandas as pd
import pandas as pd

# importing numpy as np
import numpy as np

# dictionary of lists
dict = {'First Score':[100, 90, np.nan, 95],
        'Second Score': [30, 45, 56, np.nan],
        'Third Score':[np.nan, 40, 80, 98]}

# creating a dataframe from dictionary
df = pd.DataFrame(dict)

# filling missing value using fillna()
df.fillna(0)
```

**Output:**

|   | First Score | Second Score | Third Score |
|---|-------------|--------------|-------------|
| 0 | 100.0       | 30.0         | 0.0         |
| 1 | 90.0        | 45.0         | 40.0        |
| 2 | 0.0         | 56.0         | 80.0        |
| 3 | 95.0        | 0.0          | 98.0        |

**Code #2:** Filling null values with the previous ones

---

## Python

```
# importing pandas as pd
import pandas as pd

# importing numpy as np
import numpy as np

# dictionary of lists
dict = {'First Score':[100, 90, np.nan, 95],
        'Second Score': [30, 45, 56, np.nan],
        'Third Score':[np.nan, 40, 80, 98]}

# creating a dataframe from dictionary
df = pd.DataFrame(dict)

# filling a missing value with
# previous ones
df.fillna(method='pad')
```

**Output:**

|   | First Score | Second Score | Third Score |
|---|-------------|--------------|-------------|
| 0 | 100.0       | 30.0         | NaN         |
| 1 | 90.0        | 45.0         | 40.0        |
| 2 | 90.0        | 56.0         | 80.0        |
| 3 | 95.0        | 56.0         | 98.0        |

Code #3: Filling null value with the next ones

---

## Python

```
# importing pandas as pd
import pandas as pd

# importing numpy as np
import numpy as np

# dictionary of lists
dict = {'First Score':[100, 90, np.nan, 95],
        'Second Score': [30, 45, 56, np.nan],
        'Third Score':[np.nan, 40, 80, 98]}

# creating a dataframe from dictionary
df = pd.DataFrame(dict)

# filling null value using fillna() function
df.fillna(method='bfill')
```

Output:



|   | First Score | Second Score | Third Score |
|---|-------------|--------------|-------------|
| 0 | 100.0       | 30.0         | 40.0        |
| 1 | 90.0        | 45.0         | 40.0        |
| 2 | 95.0        | 56.0         | 80.0        |
| 3 | 95.0        | NaN          | 98.0        |

#### Code #4: Filling null values in CSV File

---

### Python

```
# importing pandas package
import pandas as pd

# making data frame from csv file
data = pd.read_csv("employees.csv")

# Printing the first 10 to 24 rows of
# the data frame for visualization
data[10:25]
```

### Output

|    | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team            |
|----|------------|--------|------------|-----------------|--------|---------|-------------------|-----------------|
| 10 | Louise     | Female | 8/12/1980  | 9:01 AM         | 63241  | 15.132  | True              | NaN             |
| 11 | Julie      | Female | 10/26/1997 | 3:19 PM         | 102508 | 12.637  | True              | Legal           |
| 12 | Brandon    | Male   | 12/1/1980  | 1:08 AM         | 112807 | 17.492  | True              | Human Resources |
| 13 | Gary       | Male   | 1/27/2008  | 11:40 PM        | 109831 | 5.831   | False             | Sales           |
| 14 | Kimberly   | Female | 1/14/1999  | 7:13 AM         | 41426  | 14.543  | True              | Finance         |
| 15 | Lillian    | Female | 6/5/2016   | 6:09 AM         | 59414  | 1.256   | False             | Product         |
| 16 | Jeremy     | Male   | 9/21/2010  | 5:56 AM         | 90370  | 7.369   | False             | Human Resources |
| 17 | Shawn      | Male   | 12/7/1986  | 7:45 PM         | 111737 | 6.414   | False             | Product         |
| 18 | Diana      | Female | 10/23/1981 | 10:27 AM        | 132940 | 19.082  | False             | Client Services |
| 19 | Donna      | Female | 7/22/2010  | 3:48 AM         | 81014  | 1.894   | False             | Product         |
| 20 | Lois       | NaN    | 4/22/1995  | 7:18 PM         | 64714  | 4.934   | True              | Legal           |
| 21 | Matthew    | Male   | 9/5/1995   | 2:12 AM         | 100612 | 13.645  | False             | Marketing       |
| 22 | Joshua     | NaN    | 3/8/2012   | 1:58 AM         | 90816  | 18.816  | True              | Client Services |
| 23 | NaN        | Male   | 6/14/2012  | 4:19 PM         | 125792 | 5.042   | NaN               | NaN             |
| 24 | John       | Male   | 7/1/1992   | 10:08 PM        | 97950  | 13.873  | False             | Client Services |

Now we are going to fill all the null values in Gender column with “No Gender”

---

### Python

```
# importing pandas package
import pandas as pd

# making data frame from csv file
data = pd.read_csv("employees.csv")

# filling a null values using fillna()
data["Gender"].fillna("No Gender", inplace = True)

data
```

**Output:**

|    |          |           |            |          |        |        |       |                 |
|----|----------|-----------|------------|----------|--------|--------|-------|-----------------|
| 10 | Louise   | Female    | 8/12/1980  | 9:01 AM  | 63241  | 15.132 | True  | NaN             |
| 11 | Julie    | Female    | 10/26/1997 | 3:19 PM  | 102508 | 12.637 | True  | Legal           |
| 12 | Brandon  | Male      | 12/1/1980  | 1:08 AM  | 112807 | 17.492 | True  | Human Resources |
| 13 | Gary     | Male      | 1/27/2008  | 11:40 PM | 109831 | 5.831  | False | Sales           |
| 14 | Kimberly | Female    | 1/14/1999  | 7:13 AM  | 41426  | 14.543 | True  | Finance         |
| 15 | Lillian  | Female    | 6/5/2016   | 6:09 AM  | 59414  | 1.256  | False | Product         |
| 16 | Jeremy   | Male      | 9/21/2010  | 5:56 AM  | 90370  | 7.369  | False | Human Resources |
| 17 | Shawn    | Male      | 12/7/1986  | 7:45 PM  | 111737 | 6.414  | False | Product         |
| 18 | Diana    | Female    | 10/23/1981 | 10:27 AM | 132940 | 19.082 | False | Client Services |
| 19 | Donna    | Female    | 7/22/2010  | 3:48 AM  | 81014  | 1.894  | False | Product         |
| 20 | Lois     | No Gender | 4/22/1995  | 7:18 PM  | 64714  | 4.934  | True  | Legal           |
| 21 | Matthew  | Male      | 9/5/1995   | 2:12 AM  | 100612 | 13.645 | False | Marketing       |
| 22 | Joshua   | No Gender | 3/8/2012   | 1:58 AM  | 90816  | 18.816 | True  | Client Services |
| 23 | NaN      | Male      | 6/14/2012  | 4:19 PM  | 125792 | 5.042  | NaN   | NaN             |
| 24 | John     | Male      | 7/1/1992   | 10:08 PM | 97950  | 13.873 | False | Client Services |

**Code #5:** Filling a null values using replace() method

---

**Python**

```
# importing pandas package
import pandas as pd

# making data frame from csv file
data = pd.read_csv("employees.csv")

# Printing the first 10 to 24 rows of
# the data frame for visualization
data[10:25]
```

**Output:**

|    | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team            |
|----|------------|--------|------------|-----------------|--------|---------|-------------------|-----------------|
| 10 | Louise     | Female | 8/12/1980  | 9:01 AM         | 63241  | 15.132  | True              | NaN             |
| 11 | Julie      | Female | 10/26/1997 | 3:19 PM         | 102508 | 12.637  | True              | Legal           |
| 12 | Brandon    | Male   | 12/1/1980  | 1:08 AM         | 112807 | 17.492  | True              | Human Resources |
| 13 | Gary       | Male   | 1/27/2008  | 11:40 PM        | 109831 | 5.831   | False             | Sales           |
| 14 | Kimberly   | Female | 1/14/1999  | 7:13 AM         | 41426  | 14.543  | True              | Finance         |
| 15 | Lillian    | Female | 6/5/2016   | 6:09 AM         | 59414  | 1.256   | False             | Product         |
| 16 | Jeremy     | Male   | 9/21/2010  | 5:56 AM         | 90370  | 7.369   | False             | Human Resources |
| 17 | Shawn      | Male   | 12/7/1986  | 7:45 PM         | 111737 | 6.414   | False             | Product         |
| 18 | Diana      | Female | 10/23/1981 | 10:27 AM        | 132940 | 19.082  | False             | Client Services |
| 19 | Donna      | Female | 7/22/2010  | 3:48 AM         | 81014  | 1.894   | False             | Product         |
| 20 | Lois       | NaN    | 4/22/1995  | 7:18 PM         | 64714  | 4.934   | True              | Legal           |
| 21 | Matthew    | Male   | 9/5/1995   | 2:12 AM         | 100612 | 13.645  | False             | Marketing       |
| 22 | Joshua     | NaN    | 3/8/2012   | 1:58 AM         | 90816  | 18.816  | True              | Client Services |
| 23 | NaN        | Male   | 6/14/2012  | 4:19 PM         | 125792 | 5.042   | NaN               | NaN             |

Now we are going to replace the all Nan value in the data frame with -99 value.

## Python

```
# importing pandas package
import pandas as pd

# making data frame from csv file
data = pd.read_csv("employees.csv")

# will replace Nan value in dataframe with value -99
data.replace(to_replace = np.nan, value = -99)
```

### Output:

|    |          |        |            |          |        |        |       |                 |
|----|----------|--------|------------|----------|--------|--------|-------|-----------------|
| 10 | Louise   | Female | 8/12/1980  | 9:01 AM  | 63241  | 15.132 | True  | -99             |
| 11 | Julie    | Female | 10/26/1997 | 3:19 PM  | 102508 | 12.637 | True  | Legal           |
| 12 | Brandon  | Male   | 12/1/1980  | 1:08 AM  | 112807 | 17.492 | True  | Human Resources |
| 13 | Gary     | Male   | 1/27/2008  | 11:40 PM | 109831 | 5.831  | False | Sales           |
| 14 | Kimberly | Female | 1/14/1999  | 7:13 AM  | 41426  | 14.543 | True  | Finance         |
| 15 | Lillian  | Female | 6/5/2016   | 6:09 AM  | 59414  | 1.256  | False | Product         |
| 16 | Jeremy   | Male   | 9/21/2010  | 5:56 AM  | 90370  | 7.369  | False | Human Resources |
| 17 | Shawn    | Male   | 12/7/1986  | 7:45 PM  | 111737 | 6.414  | False | Product         |
| 18 | Diana    | Female | 10/23/1981 | 10:27 AM | 132940 | 19.082 | False | Client Services |
| 19 | Donna    | Female | 7/22/2010  | 3:48 AM  | 81014  | 1.894  | False | Product         |
| 20 | Lois     | -99    | 4/22/1995  | 7:18 PM  | 64714  | 4.934  | True  | Legal           |
| 21 | Matthew  | Male   | 9/5/1995   | 2:12 AM  | 100612 | 13.645 | False | Marketing       |
| 22 | Joshua   | -99    | 3/8/2012   | 1:58 AM  | 90816  | 18.816 | True  | Client Services |
| 23 | -99      | Male   | 6/14/2012  | 4:19 PM  | 125792 | 5.042  | -99   | -99             |
| 24 | John     | Male   | 7/1/1992   | 10:08 PM | 97950  | 13.873 | False | Client Services |

**Code #6:** Using interpolate() function to fill the missing values using linear method.

## Python

```
# importing pandas as pd
import pandas as pd

# Creating the dataframe
df = pd.DataFrame({"A": [12, 4, 5, None, 1],
                   "B": [None, 2, 54, 3, None],
                   "C": [20, 16, None, 3, 8],
                   "D": [14, 3, None, None, 6]})

# Print the dataframe
df
```

Output:

|   | A    | B    | C    | D    |
|---|------|------|------|------|
| 0 | 12.0 | NaN  | 20.0 | 14.0 |
| 1 | 4.0  | 2.0  | 16.0 | 3.0  |
| 2 | 5.0  | 54.0 | NaN  | NaN  |
| 3 | NaN  | 3.0  | 3.0  | NaN  |
| 4 | 1.0  | NaN  | 8.0  | 6.0  |

Let's interpolate the missing values using Linear method. Note that Linear method ignore the index and treat the values as equally spaced.

---

## Python

```
# to interpolate the missing values
df.interpolate(method='linear', limit_direction='forward')
```

Output:

|   | A    | B    | C    | D    |
|---|------|------|------|------|
| 0 | 12.0 | NaN  | 20.0 | 14.0 |
| 1 | 4.0  | 2.0  | 16.0 | 3.0  |
| 2 | 5.0  | 54.0 | 9.5  | 4.0  |
| 3 | 3.0  | 3.0  | 3.0  | 5.0  |
| 4 | 1.0  | 3.0  | 8.0  | 6.0  |

As we can see the output, values in the first row could not get filled as the direction of filling of values is forward and there is no previous value which could have been used in interpolation.

## Dropping missing values using dropna()

In order to drop a null values from a dataframe, we used dropna() function this function drop Rows/Columns of datasets with Null values in different ways.

**Code #1:** Dropping rows with at least 1 null value.

## Python

```
# importing pandas as pd
import pandas as pd

# importing numpy as np
import numpy as np

# dictionary of lists
dict = {'First Score':[100, 90, np.nan, 95],
        'Second Score': [30, np.nan, 45, 56],
        'Third Score':[52, 40, 80, 98],
        'Fourth Score':[np.nan, np.nan, np.nan, 65]}

# creating a dataframe from dictionary
df = pd.DataFrame(dict)

df
```

## Output

|   | First Score | Second Score | Third Score | Fourth Score |
|---|-------------|--------------|-------------|--------------|
| 0 | 100.0       | 30.0         | 52          | NaN          |
| 1 | 90.0        | NaN          | 40          | NaN          |
| 2 | NaN         | 45.0         | 80          | NaN          |
| 3 | 95.0        | 56.0         | 98          | 65.0         |

Now we drop rows with at least one Nan value (Null value)

---

## Python

```
# importing pandas as pd
import pandas as pd

# importing numpy as np
import numpy as np

# dictionary of lists
dict = {'First Score':[100, 90, np.nan, 95],
        'Second Score': [30, np.nan, 45, 56],
        'Third Score':[52, 40, 80, 98],
        'Fourth Score':[np.nan, np.nan, np.nan, 65]}

# creating a dataframe from dictionary
df = pd.DataFrame(dict)

# using dropna() function
df.dropna()
```

Output:

|   | First Score | Second Score | Third Score | Fourth Score |
|---|-------------|--------------|-------------|--------------|
| 3 | 95.0        | 56.0         | 98          | 65.0         |

Code #2: Dropping rows if all values in that row are missing.

---

## Python

```
# importing pandas as pd
import pandas as pd

# importing numpy as np
import numpy as np

# dictionary of lists
dict = {'First Score':[100, np.nan, np.nan, 95],
        'Second Score': [30, np.nan, 45, 56],
        'Third Score':[52, np.nan, 80, 98],
        'Fourth Score':[np.nan, np.nan, np.nan, 65]}

# creating a dataframe from dictionary
df = pd.DataFrame(dict)

df
```

## Output

|   | First Score | Second Score | Third Score | Fourth Score |
|---|-------------|--------------|-------------|--------------|
| 0 | 100.0       | 30.0         | 52.0        | NaN          |
| 1 | NaN         | NaN          | NaN         | NaN          |
| 2 | NaN         | 45.0         | 80.0        | NaN          |
| 3 | 95.0        | 56.0         | 98.0        | 65.0         |

Now we drop a rows whose all data is missing or contain null values(NaN)

---

## Python

```
# importing pandas as pd
import pandas as pd

# importing numpy as np
import numpy as np

# dictionary of lists
dict = {'First Score':[100, np.nan, np.nan, 95],
        'Second Score': [30, np.nan, 45, 56],
        'Third Score':[52, np.nan, 80, 98],
        'Fourth Score':[np.nan, np.nan, np.nan, 65]}

df = pd.DataFrame(dict)

# using dropna() function
df.dropna(how = 'all')
```

## Output:



|   | First Score | Second Score | Third Score | Fourth Score |
|---|-------------|--------------|-------------|--------------|
| 0 | 100.0       | 30.0         | 52.0        | NaN          |
| 2 | NaN         | 45.0         | 80.0        | NaN          |
| 3 | 95.0        | 56.0         | 98.0        | 65.0         |

**Code #3:** Dropping columns with at least 1 null value.

---

## Python

```
# importing pandas as pd
import pandas as pd

# importing numpy as np
import numpy as np

# dictionary of lists
dict = {'First Score':[100, np.nan, np.nan, 95],
        'Second Score': [30, np.nan, 45, 56],
        'Third Score':[52, np.nan, 80, 98],
        'Fourth Score':[60, 67, 68, 65]}

# creating a dataframe from dictionary
df = pd.DataFrame(dict)

df
```

## Output

|   | First Score | Second Score | Third Score | Fourth Score |
|---|-------------|--------------|-------------|--------------|
| 0 | 100.0       | 30.0         | 52.0        | 60           |
| 1 | NaN         | NaN          | NaN         | 67           |
| 2 | NaN         | 45.0         | 80.0        | 68           |
| 3 | 95.0        | 56.0         | 98.0        | 65           |

Now we drop a columns which have at least 1 missing values

---

## Python

```
# importing pandas as pd
import pandas as pd

# importing numpy as np
import numpy as np

# dictionary of lists
dict = {'First Score':[100, np.nan, np.nan, 95],
        'Second Score': [30, np.nan, 45, 56],
        'Third Score':[52, np.nan, 80, 98],
        'Fourth Score':[60, 67, 68, 65]}

# creating a dataframe from dictionary
df = pd.DataFrame(dict)

# using dropna() function
df.dropna(axis = 1)
```

Output :

## Fourth Score

0 60

1 67

2 68

3 65

**Code #4:** Dropping Rows with at least 1 null value in CSV file

---

## Python

```
# importing pandas module
import pandas as pd

# making data frame from csv file
data = pd.read_csv("employees.csv")

# making new data frame with dropped NA values
new_data = data.dropna(axis = 0, how = 'any')

new_data
```

## Output:

|     | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team                 |
|-----|------------|--------|------------|-----------------|--------|---------|-------------------|----------------------|
| 0   | Douglas    | Male   | 8/6/1993   | 12:42 PM        | 97308  | 6.945   | True              | Marketing            |
| 2   | Maria      | Female | 4/23/1993  | 11:17 AM        | 130590 | 11.858  | False             | Finance              |
| 3   | Jerry      | Male   | 3/4/2005   | 1:00 PM         | 138705 | 9.340   | True              | Finance              |
| 4   | Larry      | Male   | 1/24/1998  | 4:47 PM         | 101004 | 1.389   | True              | Client Services      |
| 5   | Dennis     | Male   | 4/18/1987  | 1:35 AM         | 115163 | 10.125  | False             | Legal                |
| 6   | Ruby       | Female | 8/17/1987  | 4:20 PM         | 65476  | 10.012  | True              | Product              |
| 8   | Angela     | Female | 11/22/2005 | 6:29 AM         | 95570  | 18.523  | True              | Engineering          |
| 9   | Frances    | Female | 8/8/2002   | 6:51 AM         | 139852 | 7.524   | True              | Business Development |
| 11  | Julie      | Female | 10/26/1997 | 3:19 PM         | 102508 | 12.637  | True              | Legal                |
| 12  | Brandon    | Male   | 12/1/1980  | 1:08 AM         | 112807 | 17.492  | True              | Human Resources      |
| 13  | Gary       | Male   | 1/27/2008  | 11:40 PM        | 109831 | 5.831   | False             | Sales                |
| 14  | Kimberly   | Female | 1/14/1999  | 7:13 AM         | 41426  | 14.543  | True              | Finance              |
| ... | ...        | ...    | ...        | ...             | ...    | ...     | ...               | ...                  |
| ... | ...        | ...    | ...        | ...             | ...    | ...     | ...               | ...                  |
| ... | ...        | ...    | ...        | ...             | ...    | ...     | ...               | ...                  |
| 993 | Tina       | Female | 5/15/1997  | 3:53 PM         | 56450  | 19.040  | True              | Engineering          |
| 994 | George     | Male   | 6/21/2013  | 5:47 PM         | 98874  | 4.479   | True              | Marketing            |
| 996 | Phillip    | Male   | 1/31/1984  | 6:30 AM         | 42392  | 19.675  | False             | Finance              |
| 997 | Russell    | Male   | 5/20/2013  | 12:39 PM        | 96914  | 1.421   | False             | Product              |
| 998 | Larry      | Male   | 4/20/2013  | 4:45 PM         | 60500  | 11.985  | False             | Business Development |
| 999 | Albert     | Male   | 5/15/2012  | 6:24 PM         | 129949 | 10.169  | True              | Sales                |

764 rows x 8 columns

Now we compare sizes of data frames so that we can come to know how many rows had at least 1 Null value

---

## Python

```
print("Old data frame length:", len(data))
print("New data frame length:", len(new_data))
print("Number of rows with at least 1 NA value: ", (len(data)-len(new_data)))
```

### Output :

```
Old data frame length: 1000
New data frame length: 764
Number of rows with at least 1 NA value:  236
```

Since the difference is 236, there were 236 rows which had at least 1 Null value in any column.

Last Updated : 09 Feb, 2023

52

## Similar Reads

1. [Drop rows from Pandas dataframe with missing values or NaN in columns](#)
2. [Count NaN or missing values in Pandas DataFrame](#)
3. [Replacing missing values using Pandas in Python](#)
4. [Check missing dates in Pandas](#)
5. [Replace missing white spaces in a string with the least frequent character using Pandas](#)
6. [Python | Working with date and time using Pandas](#)
7. [Python | Working with Pandas and XlsxWriter | Set - 1](#)
8. [Python | Working with Pandas and XlsxWriter | Set – 2](#)
9. [Python | Working with Pandas and XlsxWriter | Set – 3](#)

## Related Tutorials

1. Data Analysis Tutorial
2. Flask Tutorial
3. Natural Language Processing (NLP) Tutorial
4. Data Science for Beginners
5. Data Science With Python Tutorial

[Previous](#)

[Next](#)

### Article Contributed By :



GeeksforGeeks

### Vote for difficulty

Current difficulty : [Basic](#)

Easy

Normal

Medium

Hard

Expert

Improved By : [Akanksha\\_Rai](#), [simmytarika5](#)

Article Tags : [Python pandas-dataFrame](#), [Python pandas-missingData](#), [Python-pandas](#), [Python](#)

Practice Tags : [python](#)

Improve Article

Report Issue



GeeksforGeeks

A-143, 9th Floor, Sovereign Corporate  
Tower, Sector-136, Noida, Uttar Pradesh -  
201305

[feedback@geeksforgeeks.org](mailto:feedback@geeksforgeeks.org)

## Company

About Us  
Careers  
In Media  
Contact Us  
Terms and Conditions  
Privacy Policy  
Copyright Policy  
Third-Party Copyright Notices  
Advertise with us

## Languages

Python  
Java  
C++  
GoLang  
SQL  
R Language  
Android Tutorial

## Algorithms

Sorting  
Searching  
Greedy  
Dynamic Programming  
Pattern Searching  
Recursion  
Backtracking

## Computer Science

GATE CS Notes  
Operating Systems  
Computer Network  
Database Management System

## Explore

Job Fair For Students  
POTD: Revamped  
Python Backend LIVE  
Android App Development  
DevOps LIVE  
DSA in JavaScript

## Data Structures

Array  
String  
Linked List  
Stack  
Queue  
Tree  
Graph

## Web Development

HTML  
CSS  
JavaScript  
Bootstrap  
ReactJS  
AngularJS  
NodeJS

## Python

Python Programming Examples  
Django Tutorial  
Python Projects  
Python Tkinter

[Software Engineering](#)

[Digital Logic Design](#)

[Engineering Maths](#)

## Data Science & ML

[Data Science With Python](#)

[Data Science For Beginner](#)

[Machine Learning Tutorial](#)

[Maths For Machine Learning](#)

[Pandas Tutorial](#)

[NumPy Tutorial](#)

[NLP Tutorial](#)

[Deep Learning Tutorial](#)

## Competitive Programming

[Top DSA for CP](#)

[Top 50 Tree Problems](#)

[Top 50 Graph Problems](#)

[Top 50 Array Problems](#)

[Top 50 String Problems](#)

[Top 50 DP Problems](#)

[Top 15 Websites for CP](#)

## Interview Corner

[Company Preparation](#)

[Preparation for SDE](#)

[Company Interview Corner](#)

[Experienced Interview](#)

[Internship Interview](#)

[Competitive Programming](#)

[Aptitude](#)

## Commerce

[Accountancy](#)

[Business Studies](#)

[Microeconomics](#)

[Macroeconomics](#)

[OpenCV Python Tutorial](#)

[Python Interview Question](#)

## DevOps

[Git](#)

[AWS](#)

[Docker](#)

[Kubernetes](#)

[Azure](#)

[GCP](#)

## System Design

[What is System Design](#)

[Monolithic and Distributed SD](#)

[Scalability in SD](#)

[Databases in SD](#)

[High Level Design or HLD](#)

[Low Level Design or LLD](#)

[Top SD Interview Questions](#)

## GfG School

[CBSE Notes for Class 8](#)

[CBSE Notes for Class 9](#)

[CBSE Notes for Class 10](#)

[CBSE Notes for Class 11](#)

[CBSE Notes for Class 12](#)

[English Grammar](#)

## UPSC

[Polity Notes](#)

[Geography Notes](#)

[History Notes](#)

[Science and Technology Notes](#)

Statistics for Economics  
Indian Economic Development

Economics Notes  
Important Topics in Ethics  
UPSC Previous Year Papers

## **SSC/ BANKING**

SSC CGL Syllabus  
SBI PO Syllabus  
SBI Clerk Syllabus  
IBPS PO Syllabus  
IBPS Clerk Syllabus  
Aptitude Questions  
SSC CGL Practice Papers

## **Write & Earn**

Write an Article  
Improve an Article  
Pick Topics to Write  
Write Interview Experience  
Internships  
Video Internship

@geeksforgeeks , Some rights reserved