

▼ Reading a CSV File form Google Dirve

```
from google.colab import drive
drive.mount("/content/drive/")
```

Drive already mounted at /content/drive/; to attempt to forcibly remount, call drive.mou



```
import pandas as pd
Dataset = pd.read_csv("/content/drive/MyDrive/Mtech/Missing_Data.csv", sep = ',')
Dataset.head()
```

	State	Age	Height	Weight	Salary	Purchaged	
0	Andhra Pradesh	44.0	160.0	65.0	72000.0	No	
1	Arunachal Pradesh	27.0	165.0	70.0	48000.0	Yes	
2	Assam	30.0	167.0	90.0	54000.0	No	
3	Bihar	38.0	180.0	74.0	61000.0	No	
4	Chhattisgarh	40.0	168.0	75.0	NaN	Yes	

1. Show the first 10 rows of the Dataset

```
Dataset.head(10)
```

	State	Age	Height	Weight	Salary	Purchased
0	Andhra Pradesh	44.0	160.0	65.0	72000.0	No



2. Number of Rows and Columns in the Dataset

```
Dataset.shape
```

```
(112, 6)
```

3. Show last 6 records in the dataset

```
Dataset.tail(6)
```

	State	Age	Height	Weight	Salary	Purchased
106	Tamil Nadu	50.0	165.0	90.0	52000.0	No
107	Telangana	37.0	167.0	74.0	79000.0	Yes
108	Tripura	38.0	180.0	75.0	83000.0	No
109	Uttar Pradesh	42.0	168.0	87.0	67000.0	Yes
110	Uttarakhand	45.0	169.0	89.0	70000.0	Yes
111	West Bengal	51.0	161.0	88.0	52000.0	No



4. Missing values

```
import numpy as np
print(np.NaN==0)
```

```
False
```

```
print(np.NaN=='')
```

```
False
```

5. Size of the dataset

```
Dataset.size
```

```
672
```

6. Check whether there are any nullvalues or not

```
Dataset.isnull().head()
```

	State	Age	Height	Weight	Salary	Purchased
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	False	False	False	False
3	False	False	False	False	False	False
4	False	False	False	False	True	False

```
Dataset.isna().any()
```

```
State      False
Age        True
Height     True
Weight     True
Salary     True
Purchased  False
dtype: bool
```

```
Dataset.isna().sum()
```

```
State      0
Age        8
Height     9
Weight     4
Salary    11
Purchased  0
dtype: int64
```

7. To Find missing value to a particular Column

```
Dataset['Age'].isna().sum()
```

```
8
```

```
Dataset.count()
```

```
State      112
Age        104
Height     103
Weight     108
Salary     101
Purchased  112
dtype: int64
```

▼ Handling Missing Values

We can drop a row or cloumn with mssing values using *dropna()* function.

any : drop is any missing values are there *all* : drop if all are missing values

```
d1 = Dataset.copy()
d1.dropna(axis=0, inplace=True)
d1.head(10)
```

	State	Age	Height	Weight	Salary	Purchased	
0	Andhra Pradesh	44.0	160.0	65.0	72000.0	No	
1	Arunachal Pradesh	27.0	165.0	70.0	48000.0	Yes	
2	Assam	30.0	167.0	90.0	54000.0	No	
3	Bihar	38.0	180.0	74.0	61000.0	No	
5	Goa	35.0	169.0	87.0	58000.0	Yes	
7	Haryana	48.0	167.0	88.0	79000.0	Yes	
8	Himachal Pradesh	50.0	155.0	94.0	83000.0	No	
9	Jharkhand	37.0	159.0	80.0	67000.0	Yes	
10	Karnataka	38.0	158.0	65.0	72000.0	No	
11	Kerala	42.0	155.0	67.0	48000.0	Yes	



8. Replacing missing values

1. *fillna()* function of pandas conviniently handles missing values.
2. Replace missing values with a scalar: *c.fillna(2)*
3. *fillna()* can also be used on a particular column: *c['n'].fillna(1)*
4. using method parameter, missing values can be replaced with the values before or after them
5. *c.fillna(axis=0,method='ffill')*

```
#Forward Fill
d1 = Dataset.copy()
d1.fillna(axis=0,method='ffill',inplace=True)
d1.head(10)
```

	State	Age	Height	Weight	Salary	Purchased
0	Andhra Pradesh	44.0	160.0	65.0	72000.0	No
1	Arunachal Pradesh	27.0	165.0	70.0	48000.0	Yes
2	Assam	30.0	167.0	90.0	54000.0	No
3	Bihar	38.0	180.0	74.0	61000.0	No
4	Chhattisgarh	40.0	168.0	75.0	61000.0	Yes
5	Goa	35.0	169.0	87.0	58000.0	Yes
6	Gujarat	35.0	161.0	89.0	52000.0	No
7	Haryana	48.0	167.0	88.0	79000.0	Yes
8	Himachal Pradesh	50.0	155.0	94.0	83000.0	No
9	Jharkhand	37.0	159.0	80.0	67000.0	Yes



```
#Backward fill
```

```
d2 = Dataset.copy()
```

```
d2.fillna(axis=0,method='bfill',inplace=True)
```

```
d2.head(10)
```

	State	Age	Height	Weight	Salary	Purchased
0	Andhra Pradesh	44.0	160.0	65.0	72000.0	No
1	Arunachal Pradesh	27.0	165.0	70.0	48000.0	Yes
2	Assam	30.0	167.0	90.0	54000.0	No
3	Bihar	38.0	180.0	74.0	61000.0	No
4	Chhattisgarh	40.0	168.0	75.0	58000.0	Yes
5	Goa	35.0	169.0	87.0	58000.0	Yes
6	Gujarat	48.0	161.0	89.0	52000.0	No
7	Haryana	48.0	167.0	88.0	79000.0	Yes
8	Himachal Pradesh	50.0	155.0	94.0	83000.0	No
9	Jharkhand	37.0	159.0	80.0	67000.0	Yes



9. Replacing the missing values with Mean, Median or Mode

```
df = Dataset.copy()
```

```
x = df["Salary"].mean()
```

```
df["Salary"].fillna(x,inplace = True)
```

```
print(df.to_string())
```

```


33      West Bengal  51.0  167.0  90.0  50000.000000  Yes
56      Andhra Pradesh  44.0  160.0  65.0  72000.000000  No

```

56	Arunachal Pradesh	44.0	168.0	74.0	52000.000000	No
57	Arunachal Pradesh	27.0	168.0	75.0	79000.000000	Yes
58	Assam	30.0	169.0	87.0	83000.000000	No
59	Bihar	38.0	161.0	89.0	67000.000000	Yes
60	Chhattisgarh	40.0	167.0	88.0	72000.000000	No
61	Goa	35.0	155.0	94.0	48000.000000	Yes
62	Gujarat	NaN	159.0	80.0	54000.000000	No
63	Haryana	48.0	158.0	65.0	61000.000000	No
64	Himachal Pradesh	50.0	155.0	67.0	63722.772277	Yes
65	Jharkhand	37.0	154.0	68.0	58000.000000	Yes
66	Karnataka	38.0	158.0	70.0	52000.000000	No
67	Kerala	42.0	161.0	71.0	79000.000000	Yes
68	Madhya Pradesh	45.0	160.0	72.0	83000.000000	No
69	Maharashtra	51.0	NaN	73.0	67000.000000	Yes
70	Manipur	44.0	160.0	74.0	72000.000000	No
71	Meghalaya	27.0	165.0	75.0	48000.000000	Yes
72	Mizoram	30.0	167.0	76.0	54000.000000	No
73	Nagaland	38.0	180.0	77.0	61000.000000	No
74	Odisha	40.0	168.0	78.0	63722.772277	Yes
75	Punjab	35.0	169.0	79.0	58000.000000	Yes
76	Rajasthan	NaN	161.0	80.0	52000.000000	No
77	Sikkim	48.0	167.0	85.0	79000.000000	Yes
78	Tamil Nadu	50.0	155.0	NaN	83000.000000	No
79	Telangana	37.0	159.0	65.0	67000.000000	Yes
80	Tripura	38.0	158.0	70.0	72000.000000	No
81	Uttar Pradesh	42.0	155.0	90.0	48000.000000	Yes
82	Uttarakhand	45.0	154.0	74.0	54000.000000	No
83	West Bengal	51.0	158.0	75.0	61000.000000	No
84	Andhra Pradesh	44.0	161.0	87.0	63722.772277	Yes
85	Arunachal Pradesh	27.0	160.0	89.0	58000.000000	Yes
86	Assam	30.0	NaN	88.0	52000.000000	No
87	Bihar	38.0	160.0	94.0	79000.000000	Yes
88	Chhattisgarh	40.0	165.0	80.0	83000.000000	No
89	Goa	35.0	167.0	65.0	67000.000000	Yes
90	Gujarat	NaN	180.0	67.0	72000.000000	No
91	Haryana	48.0	168.0	68.0	48000.000000	Yes
92	Himachal Pradesh	50.0	169.0	70.0	54000.000000	No
93	Jharkhand	37.0	161.0	71.0	61000.000000	No
94	Karnataka	38.0	167.0	72.0	63722.772277	Yes
95	Kerala	42.0	155.0	73.0	58000.000000	Yes
96	Madhya Pradesh	45.0	159.0	74.0	52000.000000	No
97	Maharashtra	51.0	158.0	75.0	79000.000000	Yes
98	Manipur	44.0	155.0	76.0	83000.000000	No
99	Meghalaya	27.0	154.0	77.0	67000.000000	Yes
100	Mizoram	30.0	158.0	78.0	72000.000000	No
101	Nagaland	38.0	161.0	79.0	48000.000000	Yes
102	Odisha	40.0	160.0	80.0	54000.000000	No
103	Punjab	35.0	NaN	85.0	61000.000000	No
104	Rajasthan	NaN	NaN	65.0	63722.772277	Yes
105	Sikkim	48.0	160.0	70.0	58000.000000	Yes
106	Tamil Nadu	50.0	165.0	90.0	52000.000000	No
107	Telangana	37.0	167.0	74.0	79000.000000	Yes
108	Tripura	38.0	180.0	75.0	83000.000000	No
109	Uttar Pradesh	42.0	168.0	87.0	67000.000000	Yes
110	Uttarakhand	45.0	169.0	89.0	70000.000000	Yes
111	West Bengal	51.0	161.0	88.0	52000.000000	No



```
X = Dataset.drop(['State', 'Purchased'], axis = 1)
X.head(10)
```

	Age	Height	Weight	Salary	
0	44.0	160.0	65.0	72000.0	
1	27.0	165.0	70.0	48000.0	
2	30.0	167.0	90.0	54000.0	
3	38.0	180.0	74.0	61000.0	
4	40.0	168.0	75.0	NaN	
5	35.0	169.0	87.0	58000.0	
6	NaN	161.0	89.0	52000.0	
7	48.0	167.0	88.0	79000.0	
8	50.0	155.0	94.0	83000.0	
9	37.0	159.0	80.0	67000.0	

```
from sklearn.impute import SimpleImputer
SM = SimpleImputer(missing_values = np.nan, strategy = 'mean')
SM.fit(X)
#Repeating the missing value using transform method
X = SM.transform(X)
df = pd.DataFrame(X, columns = ['Age', 'Height', 'Weight', 'Salary'])
print(df.head(20))
```

	Age	Height	Weight	Salary
0	44.000000	160.000000	65.0	72000.000000
1	27.000000	165.000000	70.0	48000.000000
2	30.000000	167.000000	90.0	54000.000000
3	38.000000	180.000000	74.0	61000.000000
4	40.000000	168.000000	75.0	63722.772277
5	35.000000	169.000000	87.0	58000.000000
6	40.384615	161.000000	89.0	52000.000000
7	48.000000	167.000000	88.0	79000.000000
8	50.000000	155.000000	94.0	83000.000000
9	37.000000	159.000000	80.0	67000.000000
10	38.000000	158.000000	65.0	72000.000000
11	42.000000	155.000000	67.0	48000.000000
12	45.000000	154.000000	68.0	54000.000000
13	51.000000	158.000000	70.0	61000.000000
14	44.000000	161.000000	71.0	63722.772277
15	27.000000	160.000000	72.0	58000.000000
16	30.000000	162.640777	73.0	52000.000000
17	38.000000	162.640777	74.0	79000.000000

```
18 40.000000 162.640777 75.0 83000.000000
19 35.000000 160.000000 76.0 67000.000000
```

✓

0s

completed at 4:38 PM

×