```
import pandas as pd
import numpy as np
```

```
#check for missing values by using isnull() or notnull()
df=pd.DataFrame(np.random.randn(5, 3), index=['a', 'c', 'e', 'f','h'],columns=['one', 'two', 'three'])
df=df.reindex(['a', 'b', 'c', 'd', 'e', 'f','g','h'])
print(df['one'].isnull())  #or notnull()
```

```
a    False
b     True
c    False
d     True
e    False
f    False
g     True
h    False
Name: one, dtype: bool
```

```
#Missing values:
df=pd.DataFrame(np.random.randn(5, 3), index=['a', 'c', 'e', 'f','h'],columns=['one', 'two', 'three'])
print(df)
df=df.reindex(['a', 'b', 'c', 'd', 'e', 'f','g','h'])
print(df)
```

```
          one       two     three
a  1.136327 -0.673291 -1.000880
c  0.964375 -0.273273  0.394106
e  0.099469 -0.282085 -1.886971
f  1.360555 -0.612641 -0.659640
h  0.354998  0.267584  0.773585
          one       two     three
a  1.136327 -0.673291 -1.000880
b       NaN       NaN       NaN
c  0.964375 -0.273273  0.394106
d       NaN       NaN       NaN
e  0.099469 -0.282085 -1.886971
f  1.360555 -0.612641 -0.659640
g       NaN       NaN       NaN
h  0.354998  0.267584  0.773585
```

```
#Replacing the missing values: 2 approaches
#  1]Replacing the missing values by 0
df=pd.DataFrame(np.random.randn(5, 3), index=['a', 'c', 'e', 'f','h'],columns=['one', 'two', 'three'])
df=df.reindex(['a', 'b', 'c', 'd', 'e', 'f','g','h'])
print(df)
print("Nan replaced with '0': ")
print(df.fillna(0))
```

```
          one       two     three
a -0.562780  1.091774  0.063759
b       NaN       NaN       NaN
```

```
c  0.440146 -2.112609 -0.063445
d       NaN       NaN       NaN
e -0.249815 -1.233915 -1.498123
f -0.672515  1.796693 -0.201372
g       NaN       NaN       NaN
h -0.509008  1.334055 -0.026108
Nan replaced with '0':
        one       two     three
a -0.562780  1.091774  0.063759
b  0.000000  0.000000  0.000000
c  0.440146 -2.112609 -0.063445
d  0.000000  0.000000  0.000000
e -0.249815 -1.233915 -1.498123
f -0.672515  1.796693 -0.201372
g  0.000000  0.000000  0.000000
h -0.509008  1.334055 -0.026108
```

```
#pad - Forwrad Fill
# bfill-Backward Fill
# Forward filling:"pad"
df=pd.DataFrame(np.random.randn(5, 3), index=['a', 'c', 'e', 'f','h'],columns=['one', 'two', 'three'])
df=df.reindex(['a', 'b', 'c', 'd', 'e', 'f','g','h'])
print(df)
print('----------')
print(df.fillna(method='pad'))
```

```
        one       two     three
a -0.971379  0.460897  1.019841
b       NaN       NaN       NaN
c -0.154695 -0.568130  1.483092
d       NaN       NaN       NaN
e  0.648148 -0.550359  0.064491
f  0.744508 -0.236856  0.084349
g       NaN       NaN       NaN
h  0.109584  0.738792 -0.895809
----------
        one       two     three
a -0.971379  0.460897  1.019841
b -0.971379  0.460897  1.019841
c -0.154695 -0.568130  1.483092
d -0.154695 -0.568130  1.483092
e  0.648148 -0.550359  0.064491
f  0.744508 -0.236856  0.084349
g  0.744508 -0.236856  0.084349
h  0.109584  0.738792 -0.895809
```

```
#Backward filling - "bfill"
df=pd.DataFrame(np.random.randn(5, 3), index=['a', 'c', 'e', 'f','h'],columns=['one', 'two', 'three'])
df=df.reindex(['a', 'b', 'c', 'd', 'e', 'f','g','h'])
print(df)
print('----------')
print(df.fillna(method='bfill'))
```

```
        one       two     three
a  0.580676  1.220967 -0.468235
```

```
b       NaN       NaN       NaN
c -0.716771  0.390296 -0.070266
d       NaN       NaN       NaN
e -0.927787  0.836872 -0.555422
f  0.679473  0.562773  0.303334
g       NaN       NaN       NaN
h -0.285928 -0.932914 -0.075519
----------
        one       two     three
a  0.580676  1.220967 -0.468235
b -0.716771  0.390296 -0.070266
c -0.716771  0.390296 -0.070266
d -0.927787  0.836872 -0.555422
e -0.927787  0.836872 -0.555422
f  0.679473  0.562773  0.303334
g -0.285928 -0.932914 -0.075519
h -0.285928 -0.932914 -0.075519
```

```
#2] dropping the missing values:
df=pd.DataFrame(np.random.randn(5, 3), index=['a', 'c', 'e', 'f','h'],columns=['one', 'two', 'three'])
print(df)
df=df.reindex(['a', 'b', 'c', 'd', 'e', 'f','g','h'])
print(df)
print(df.dropna())
```

```
        one       two     three
a -0.834667 -0.971199  1.503478
c -0.282443  0.947684  2.592310
e  0.221703  0.248083 -0.328506
f  1.223284  0.571591  0.134609
h  0.686904 -0.291878  0.176058
        one       two     three
a -0.834667 -0.971199  1.503478
b       NaN       NaN       NaN
c -0.282443  0.947684  2.592310
d       NaN       NaN       NaN
e  0.221703  0.248083 -0.328506
f  1.223284  0.571591  0.134609
g       NaN       NaN       NaN
h  0.686904 -0.291878  0.176058
        one       two     three
a -0.834667 -0.971199  1.503478
c -0.282443  0.947684  2.592310
e  0.221703  0.248083 -0.328506
f  1.223284  0.571591  0.134609
h  0.686904 -0.291878  0.176058
```

DATA PREPROCESSING

```
df=pd.read_csv("/2,1 dataset titanic.csv")
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12 entries, 0 to 11
Data columns (total 5 columns):
```

```
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Speed (mph)  12 non-null     float64
 1   Driver       12 non-null     object
 2   Car          12 non-null     object
 3   Engine       12 non-null     object
 4   Date         12 non-null     object
dtypes: float64(1), object(4)
memory usage: 608.0+ bytes
```

```
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

```
#drop the columns
cols=['Name', 'Ticket', 'Cabin']
df=df.drop(cols,axis=1)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 9 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Sex          891 non-null    object
 4   Age          714 non-null    float64
 5   SibSp        891 non-null    int64
 6   Parch        891 non-null    int64
 7   Fare         891 non-null    float64
 8   Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(2)
memory usage: 62.8+ KB
```

```
#drop the rows having no values
df=df.dropna()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 712 entries, 0 to 890
Data columns (total 9 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
```

```
 0   PassengerId  712 non-null    int64
 1   Survived     712 non-null    int64
 2   Pclass       712 non-null    int64
 3   Sex          712 non-null    object
 4   Age          712 non-null    float64
 5   SibSp        712 non-null    int64
 6   Parch        712 non-null    int64
 7   Fare         712 non-null    float64
 8   Embarked     712 non-null    object
dtypes: float64(2), int64(5), object(2)
memory usage: 55.6+ KB
```

```
#Creating Dummies   #filled with 1 and 0
dummies=[]
cols=['Pclass', 'Sex', 'Embarked']
for col in cols:
  dummies.append(pd.get_dummies(df[col]))
print(df)
```

```
     PassengerId  Survived  Pclass     Sex   Age  SibSp  Parch     Fare  \
0              1         0       3    male  22.0      1      0   7.2500
1              2         1       1  female  38.0      1      0  71.2833
2              3         1       3  female  26.0      0      0   7.9250
3              4         1       1  female  35.0      1      0  53.1000
4              5         0       3    male  35.0      0      0   8.0500
..           ...       ...     ...     ...   ...    ...    ...      ...
885          886         0       3  female  39.0      0      5  29.1250
886          887         0       2    male  27.0      0      0  13.0000
887          888         1       1  female  19.0      0      0  30.0000
889          890         1       1    male  26.0      0      0  30.0000
890          891         0       3    male  32.0      0      0   7.7500

    Embarked  1  ...  Q  S  1  2  3  female  male  C  Q  S
0          S  0  ...  0  1  0  0  1       0     1  0  0  1
1          C  1  ...  0  0  1  0  0       1     0  1  0  0
2          S  0  ...  0  1  0  0  1       1     0  0  0  1
3          S  1  ...  0  1  1  0  0       1     0  0  0  1
4          S  0  ...  0  1  0  0  1       0     1  0  0  1
..       ... ..  ... .. .. .. .. ..     ...   ... .. .. ..
885        Q  0  ...  1  0  0  0  1       1     0  0  1  0
886        S  0  ...  0  1  0  1  0       0     1  0  0  1
887        S  1  ...  0  1  1  0  0       1     0  0  0  1
889        C  1  ...  0  0  1  0  0       0     1  1  0  0
890        Q  0  ...  1  0  0  0  1       0     1  0  1  0

[712 rows x 25 columns]
```

```
#transfer the 8th column
titanic_dummies=pd.concat(dummies,axis=1)
print(df)
```

```
     PassengerId  Survived  Pclass     Sex   Age  SibSp  Parch     Fare  \
0              1         0       3    male  22.0      1      0   7.2500
1              2         1       1  female  38.0      1      0  71.2833
2              3         1       3  female  26.0      0      0   7.9250
3              4         1       1  female  35.0      1      0  53.1000
4              5         0       3    male  35.0      0      0   8.0500
..           ...       ...     ...     ...   ...    ...    ...      ...
```

```
      885          886         0       3  female  39.0      0      5  29.1250
      886          887         0       2    male  27.0      0      0  13.0000
      887          888         1       1  female  19.0      0      0  30.0000
      889          890         1       1    male  26.0      0      0  30.0000
      890          891         0       3    male  32.0      0      0   7.7500

          Embarked  1  ...  Q  S  1  2  3  female  male  C  Q  S
      0          S  0  ...  0  1  0  0  1       0     1  0  0  1
      1          C  1  ...  0  0  1  0  0       1     0  1  0  0
      2          S  0  ...  0  1  0  0  1       1     0  0  0  1
      3          S  1  ...  0  1  1  0  0       1     0  0  0  1
      4          S  0  ...  0  1  0  0  1       0     1  0  0  1
      ..       ... ..  ... .. .. .. .. ..     ...   ... .. .. ..
      885        Q  0  ...  1  0  0  0  1       1     0  0  1  0
      886        S  0  ...  0  1  0  1  0       0     1  0  0  1
      887        S  1  ...  0  1  1  0  0       1     0  0  0  1
      889        C  1  ...  0  0  1  0  0       0     1  1  0  0
      890        Q  0  ...  1  0  0  0  1       0     1  0  1  0

      [712 rows x 25 columns]


    #Concatenate the values with dataframe
    df=pd.concat((df,titanic_dummies),axis=1)
    print(df)

           PassengerId  Survived  Pclass     Sex   Age  SibSp  Parch     Fare  \
      0              1         0       3    male  22.0      1      0   7.2500
      1              2         1       1  female  38.0      1      0  71.2833
      2              3         1       3  female  26.0      0      0   7.9250
      3              4         1       1  female  35.0      1      0  53.1000
      4              5         0       3    male  35.0      0      0   8.0500
      ..           ...       ...     ...     ...   ...    ...    ...    ...      ...
      885          886         0       3  female  39.0      0      5  29.1250
      886          887         0       2    male  27.0      0      0  13.0000
      887          888         1       1  female  19.0      0      0  30.0000
      889          890         1       1    male  26.0      0      0  30.0000
      890          891         0       3    male  32.0      0      0   7.7500

          Embarked  1  ...  Q  S  1  2  3  female  male  C  Q  S
      0          S  0  ...  0  1  0  0  1       0     1  0  0  1
      1          C  1  ...  0  0  1  0  0       1     0  1  0  0
      2          S  0  ...  0  1  0  0  1       1     0  0  0  1
      3          S  1  ...  0  1  1  0  0       1     0  0  0  1
      4          S  0  ...  0  1  0  0  1       0     1  0  0  1
      ..       ... ..  ... .. .. .. .. ..     ...   ... .. .. ..
      885        Q  0  ...  1  0  0  0  1       1     0  0  1  0
      886        S  0  ...  0  1  0  1  0       0     1  0  0  1
      887        S  1  ...  0  1  1  0  0       1     0  0  0  1
      889        C  1  ...  0  0  1  0  0       0     1  1  0  0
      890        Q  0  ...  1  0  0  0  1       0     1  0  1  0

      [712 rows x 25 columns]


    #remove the unwanted cols bcz it do not impact output
    df=df.drop(['Pclass', 'Sex', 'Embarked'],axis=1)
    print(df)

           PassengerId  Survived   Age  SibSp  Parch     Fare  1  2  3  female  ...  \
      0              1         0  22.0      1      0   7.2500  0  0  1       0  ...
```

```
1             2         1  38.0      1      0  71.2833  1  0  0       1  ...
2             3         1  26.0      0      0   7.9250  0  0  1       1  ...
3             4         1  35.0      1      0  53.1000  1  0  0       1  ...
4             5         0  35.0      0      0   8.0500  0  0  1       0  ...
..          ...       ...   ...    ...    ...      ... .. .. ..     ...  ...
885         886         0  39.0      0      5  29.1250  0  0  1       1  ...
886         887         0  27.0      0      0  13.0000  0  1  0       0  ...
887         888         1  19.0      0      0  30.0000  1  0  0       1  ...
889         890         1  26.0      0      0  30.0000  1  0  0       0  ...
890         891         0  32.0      0      0   7.7500  0  0  1       0  ...

     Q  S  1  2  3  female  male  C  Q  S
0    0  1  0  0  1       0     1  0  0  1
1    0  0  1  0  0       1     0  1  0  0
2    0  1  0  0  1       1     0  0  0  1
3    0  1  1  0  0       1     0  0  0  1
4    0  1  0  0  1       0     1  0  0  1
..  .. .. .. .. ..     ...   ... .. .. ..
885  1  0  0  0  1       1     0  0  1  0
886  0  1  0  1  0       0     1  0  0  1
887  0  1  1  0  0       1     0  0  0  1
889  0  0  1  0  0       0     1  1  0  0
890  1  0  0  0  1       0     1  0  1  0

[712 rows x 22 columns]
```

```
#normalization
from sklearn.preprocessing import MinMaxScaler
data=[[-1,2],[_0,5,6],[0,10],[1,18]]
scaler=MinMaxScaler()
print(Scaler.fit(data))
print('--------')
MinMaxScaler()
print(scaler.data_max_)
print('---------')
print(scaler.transform(data))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-11-f71227dc8743> in <cell line: 3>()
      1 #normalization
      2 from sklearn.preprocessing import MinMaxScaler
----> 3 data=[[-1,2],[_0,5,6],[0,10],[1,18]]
      4 scaler=MinMaxScaler()
      5 print(Scaler.fit(data))

NameError: name '_0' is not defined
```