More Important Functions

```
# value_counts
# sort_values
# rank
# sort index
# set index
# rename index -> rename
# reset index
# unique & nunique
# isnull/notnull/hasnans
# dropna
# fillna
# drop_duplicates
# drop
# apply
# isin
# corr
# nlargest -> nsmallest
# insert
# copy
```

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

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

```
# value_counts(series and dataframe)
```

```
marks = pd.DataFrame([
    [100,80,10],
    [90,70,7],
    [120,100,14],
    [80,70,14],
    [80,70,14]
],columns=['iq','marks','package'])

marks
```

```
marks.value_counts()
```

```
ipl = pd.read_csv('ipl-matches.csv')
ipl[~ipl['MatchNumber'].str.isdigit()]['Player_of_Match'].value_counts()
```

```
# find which player has won most potm -> in finals and qualifiers
```

```
# Toss decision plot
ipl['TossDecision'].value_counts().plot(kind='pie')
```

```
# how many matches each team has played
(ipl['Team2'].value_counts() + ipl['Team1'].value_counts()).sort_values(ascending=False)
```

```
# sort_values(series and dataframe) -> ascending -> na_position -> inplace -> multiple cols
```

```
x = pd.Series([12,14,1,56,89])
x
```

```
x.sort_values(ascending=False)
```

```
movies = pd.read_csv('movies.csv')
movies.head(4)
```

```
movies.sort_values('title_x',ascending=False)
```

```
students = pd.DataFrame(
    {
        'name':['nitish','ankit','rupesh',np.nan,'mrityunjay',np.nan,'rishabh',np.nan,'aditya',np.nan],
        'college':['bit','iit','vit',np.nan,np.nan,'vlsi','ssit',np.nan,np.nan,'git'],
        'branch':['eee','it','cse',np.nan,'me','ce','civ','cse','bio',np.nan],
        'cgpa':[6.66,8.25,6.41,np.nan,5.6,9.0,7.4,10,7.4,np.nan],
        'package':[4,5,6,np.nan,6,7,8,9,np.nan,np.nan]

    }
)

students
```

```
students.sort_values('name',na_position='first',ascending=False,inplace=True)
```

```
students
```

```
movies.sort_values(['year_of_release','title_x'],ascending=[True,False])
```

```
# rank(series)
batsman = pd.read_csv('batsman_runs_ipl.csv')
batsman.head()
```

```
batsman['batting_rank'] = batsman['batsman_run'].rank(ascending=False)
batsman.sort_values('batting_rank')
```

```
# sort_index(series and dataframe)
```

```
marks = {
    'maths':67,
    'english':57,
    'science':89,
    'hindi':100
}

marks_series = pd.Series(marks)
marks_series
```

```
marks_series.sort_index(ascending=False)
```

```
movies.sort_index(ascending=False)
```

```
# set_index(dataframe) -> inplace
batsman.set_index('batter',inplace=True)
```

```
batsman
```

```
# reset_index(series + dataframe) -> drop parameter
batsman.reset_index(inplace=True)
```

```
batsman
```

```
# how to replace existing index without loosing
batsman.reset_index().set_index('batting_rank')
```

```
# series to dataframe using reset_index
marks_series.reset_index()
```

```
# rename(dataframe) -> index


movies.set_index('title_x',inplace=True)


movies.rename(columns={'imdb_id':'imdb','poster_path':'link'},inplace=True)


movies.rename(index={'Uri: The Surgical Strike':'Uri','Battalion 609':'Battalion'})


# unique(series)
temp = pd.Series([1,1,2,2,3,3,4,4,5,5,np.nan,np.nan])
print(temp)


len(temp.unique())


temp.nunique()


len(ipl['Season'].unique())


# nunique(series + dataframe) -> does not count nan -> dropna parameter
ipl['Season'].nunique()


# isnull(series + dataframe)
students['name'][students['name'].isnull()]


# notnull(series + dataframe)
students['name'][students['name'].notnull()]


# hasnans(series)
students['name'].hasnans


students


students.isnull()


students.notnull()


# dropna(series + dataframe) -> how parameter -> works like or
students['name'].dropna()


students


students.dropna(how='any')


students.dropna(how='all')


students.dropna(subset=['name'])


students.dropna(subset=['name','college'])


students.dropna(inplace=True)


# fillna(series + dataframe)
students['name'].fillna('unknown')


students


students['package'].fillna(students['package'].mean())


students['name'].fillna(method='bfill')
```

```
# drop_duplicates(series + dataframe) -> works like and -> duplicated()


temp = pd.Series([1,1,1,2,3,3,4,4])
temp.drop_duplicates()


marks.drop_duplicates(keep='last')


# find the last match played by virat kohli in Delhi
ipl['all_players'] = ipl['Team1Players'] + ipl['Team2Players']
ipl.head()


def did_kohli_play(players_list):
  return 'V Kohli' in players_list


ipl['did_kohli_play'] = ipl['all_players'].apply(did_kohli_play)
ipl[(ipl['City'] == 'Delhi') & (ipl['did_kohli_play'] == True)].drop_duplicates(subset=['City','did_kohli_play'],keep='first')


students.drop_duplicates()


# drop(series + dataframe)
temp = pd.Series([10,2,3,16,45,78,10])
temp


temp.drop(index=[0,6])


students


students.drop(columns=['branch','cgpa'],inplace=True)


students.set_index('name').drop(index=['nitish','aditya'])


# apply(series + dataframe)
temp = pd.Series([10,20,30,40,50])

temp


def sigmoid(value):
  return 1/1+np.exp(-value)


temp.apply(sigmoid)


points_df = pd.DataFrame(
    {
        '1st point':[(3,4),(-6,5),(0,0),(-10,1),(4,5)],
        '2nd point':[(-3,4),(0,0),(2,2),(10,10),(1,1)]
    }
)

points_df


def euclidean(row):
  pt_A = row['1st point']
  pt_B = row['2nd point']

  return ((pt_A[0] - pt_B[0])**2 + (pt_A[1] - pt_B[1])**2)**0.5


points_df['distance'] = points_df.apply(euclidean,axis=1)
points_df


# isin(series)


# corr
```

```
# nlargest and nsmallest(series and dataframe)

# insert(dataframe)


# copy(series + dataframe)
```