**Deleting duplicate indexes in a dataframe:**

**Syntax:**

DataFrame.drop\_duplicates(subset=None, keep='first', inplace=False)

subset : column label or sequence of labels, optional. [Only consider certain columns for identifying duplicates, by default use all of the columns]

keep : {‘first’, ‘last’, False}, default ‘first’

first : Drop duplicates except for the first occurrence.

last : Drop duplicates except for the last occurrence.

False : Drop all duplicates.

inplace : boolean, default False.

Whether to drop duplicates in place or to return a copy

df = pd.DataFrame({'userid':[1,1,1,1, 2,2,2],

'itemid':[1,1,3,4, 1,2,3] })

print(df)

o/p:

itemid userid

0 1 1

1 1 1

2 3 1

3 4 1

4 1 2

5 2 2

6 3 2

To drop duplicates in itemid column:

print(df.drop\_duplicates(subset='itemid'))

o/p:

itemid userid

0 1 1

2 3 1

3 4 1

5 2 2

To drop duplicates in useid column:

print(df.drop\_duplicates(subset='userid'))

o/p:

itemid userid

0 1 1

4 1 2

To drop duplicates in both columns:

print(df.drop\_duplicates(subset=['userid', 'itemid']))

o/p:

itemid userid

0 1 1

2 3 1

3 4 1

4 1 2

5 2 2

6 3 2

The following syntax is equivalent to the previous one. When you don’t provide any columns, subset will by default consider all the columns:

print(df.drop\_duplicates(subset=None))

**inplace:**

**inplace=True** returns None

print(df.drop\_duplicates(subset=None, inplace=True))

o/p:

None

**inplace=False** returns a copy of the object with the operation performed.

print(df.drop\_duplicates(subset = None, inplace=False))

o/p:

itemid userid

0 1 1

2 3 1

3 4 1

4 1 2

5 2 2

6 3 2

**Resetting Indexes:**

In order to reset an index we use df.reset\_index() with inplace=True option. This will create a new index to the dataframe starting from 0.

df = pd.DataFrame(data=np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9], [40, 50, 60], [23, 35, 37]]),

index= [2.5, 12.6, 4.8, 4.8, 2.5],

columns=[48, 49, 50])

print(df)

o/p:

48 49 50

2.5 1 2 3

12.6 4 5 6

4.8 7 8 9

4.8 40 50 60

2.5 23 35 37

df.reset\_index(inplace=True)

print(df)

o/p:

index 48 49 50

0 2.5 1 2 3

1 12.6 4 5 6

2 4.8 7 8 9

3 4.8 40 50 60

4 2.5 23 35 37

**Renaming Columns:**

df = pd.DataFrame([[1,2,3], [4,5,6], [7,8,9]], columns=['A','B','C'])

print(df)

o/p:

A B C

0 1 2 3

1 4 5 6

2 7 8 9

**To rename a single column:**

df.rename(columns= {'A': 'Col\_1'}, inplace=True)

print(df)

o/p:

Col\_1 B C

0 1 2 3

1 4 5 6

2 7 8 9

**To rename multiple columns:**

new\_cols = {'A': 'Col\_1', 'B': 'Col\_2', 'C':'Col\_3'}

df.rename(columns=new\_cols, inplace=True)

print(df)

o/p:

Col\_1 Col\_2 Col\_3

0 1 2 3

1 4 5 6

2 7 8 9

**Renaming rows:**

df.rename(index = {0:'row\_1', 1:'row\_2', 2:'row\_3'}, inplace=True)

print(df)

o/p:

A B C

row\_1 1 2 3

row\_2 4 5 6

row\_3 7 8 9