datacleaning

November 1, 2024

[1]:
```
# import the pandas library
import pandas as pd
import numpy as np

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['one'].median())
print (df['one'].isnull())
#Total missing value for each attribute
print (df.isnull().sum())
#any missing values?
print (df['one'].isnull().values.any())
#Total no. of missing values
print (df.isnull().sum().sum())
```

```
        one       two     three
a  0.848768 -0.128940  0.578229
c -2.804692  1.306723  0.656576
e  1.042466 -0.982625  0.023920
f -0.329088 -1.381245  1.210031
h  2.104977 -0.764836  0.975284
        one       two     three
a  0.848768 -0.128940  0.578229
b       NaN       NaN       NaN
c -2.804692  1.306723  0.656576
d       NaN       NaN       NaN
e  1.042466 -0.982625  0.023920
f -0.329088 -1.381245  1.210031
g       NaN       NaN       NaN
h  2.104977 -0.764836  0.975284
0.8487681538621735
a    False
b     True
c    False
```

```
d     True
e    False
f    False
g     True
h    False
Name: one, dtype: bool
one      3
two      3
three    3
dtype: int64
True
9
```

[2]:
```
print ("NaN replaced with '0':")
print( df.fillna(0))
```

```
NaN replaced with '0':
        one       two     three
a  0.848768 -0.128940  0.578229
b  0.000000  0.000000  0.000000
c -2.804692  1.306723  0.656576
d  0.000000  0.000000  0.000000
e  1.042466 -0.982625  0.023920
f -0.329088 -1.381245  1.210031
g  0.000000  0.000000  0.000000
h  2.104977 -0.764836  0.975284
```

[3]:
```
print(df)
print( df.fillna(method='pad'))
```

```
        one       two     three
a  0.848768 -0.128940  0.578229
b       NaN       NaN       NaN
c -2.804692  1.306723  0.656576
d       NaN       NaN       NaN
e  1.042466 -0.982625  0.023920
f -0.329088 -1.381245  1.210031
g       NaN       NaN       NaN
h  2.104977 -0.764836  0.975284
        one       two     three
a  0.848768 -0.128940  0.578229
b  0.848768 -0.128940  0.578229
c -2.804692  1.306723  0.656576
d -2.804692  1.306723  0.656576
e  1.042466 -0.982625  0.023920
f -0.329088 -1.381245  1.210031
g -0.329088 -1.381245  1.210031
h  2.104977 -0.764836  0.975284
```

```
C:\Users\Prateek\AppData\Local\Temp\ipykernel_15920\1346297352.py:2:
FutureWarning: DataFrame.fillna with 'method' is deprecated and will raise in a
future version. Use obj.ffill() or obj.bfill() instead.
  print( df.fillna(method='pad'))
```

[4]:
```
print(df)
print( df.fillna(method='bfill'))
```

```
        one       two     three
a  0.848768 -0.128940  0.578229
b       NaN       NaN       NaN
c -2.804692  1.306723  0.656576
d       NaN       NaN       NaN
e  1.042466 -0.982625  0.023920
f -0.329088 -1.381245  1.210031
g       NaN       NaN       NaN
h  2.104977 -0.764836  0.975284
        one       two     three
a  0.848768 -0.128940  0.578229
b -2.804692  1.306723  0.656576
c -2.804692  1.306723  0.656576
d  1.042466 -0.982625  0.023920
e  1.042466 -0.982625  0.023920
f -0.329088 -1.381245  1.210031
g  2.104977 -0.764836  0.975284
h  2.104977 -0.764836  0.975284
```

```
C:\Users\Prateek\AppData\Local\Temp\ipykernel_15920\190117098.py:2:
FutureWarning: DataFrame.fillna with 'method' is deprecated and will raise in a
future version. Use obj.ffill() or obj.bfill() instead.
  print( df.fillna(method='bfill'))
```

[5]:
```
print(df)
print( df.dropna())
```

```
        one       two     three
a  0.848768 -0.128940  0.578229
b       NaN       NaN       NaN
c -2.804692  1.306723  0.656576
d       NaN       NaN       NaN
e  1.042466 -0.982625  0.023920
f -0.329088 -1.381245  1.210031
g       NaN       NaN       NaN
h  2.104977 -0.764836  0.975284
        one       two     three
a  0.848768 -0.128940  0.578229
c -2.804692  1.306723  0.656576
e  1.042466 -0.982625  0.023920
```

```
f -0.329088 -1.381245  1.210031
h  2.104977 -0.764836  0.975284
```

[6]:
```
#Interpolation of immediate data before and after it (average is taken)

print(df.interpolate())
```

```
        one       two     three
a  0.848768 -0.128940  0.578229
b -0.977962  0.588891  0.617403
c -2.804692  1.306723  0.656576
d -0.881113  0.162049  0.340248
e  1.042466 -0.982625  0.023920
f -0.329088 -1.381245  1.210031
g  0.887945 -1.073041  1.092658
h  2.104977 -0.764836  0.975284
```

[7]:
```
import pandas as pd

df = pd.read_csv("loan_data_set.csv")       #paste entire file path
df.head()
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
Cell In[7], line 3
      1 import pandas as pd
----> 3 df = pd.read_csv("loan_data_set.csv")       #paste entire file path
      4 df.head()

File c:
 ↪\Users\Prateek\AppData\Local\Programs\Python\Python311\Lib\site-packages\pandas\io\parsers
 ↪py:1026, in read_csv(filepath_or_buffer, sep, delimiter, header, names,␣
 ↪index_col, usecols, dtype, engine, converters, true_values, false_values,␣
 ↪skipinitialspace, skiprows, skipfooter, nrows, na_values, keep_default_na,␣
 ↪na_filter, verbose, skip_blank_lines, parse_dates, infer_datetime_format,␣
 ↪keep_date_col, date_parser, date_format, dayfirst, cache_dates, iterator,␣
 ↪chunksize, compression, thousands, decimal, lineterminator, quotechar,␣
 ↪quoting, doublequote, escapechar, comment, encoding, encoding_errors, dialect␣
 ↪on_bad_lines, delim_whitespace, low_memory, memory_map, float_precision,␣
 ↪storage_options, dtype_backend)
   1013 kwds_defaults = _refine_defaults_read(
   1014     dialect,
   1015     delimiter,
   (…)
   1022     dtype_backend=dtype_backend,
   1023 )
   1024 kwds.update(kwds_defaults)
-> 1026 return _read(filepath_or_buffer, kwds)
```

```
File c:
 ↪\Users\Prateek\AppData\Local\Programs\Python\Python311\Lib\site-packages\pand as\io\parsers\
 ↪py:620, in _read(filepath_or_buffer, kwds)
    617 _validate_names(kwds.get("names", None))
    619 # Create the parser.
--> 620 parser = TextFileReader(filepath_or_buffer, **kwds)
    622 if chunksize or iterator:
    623     return parser

File c:
 ↪\Users\Prateek\AppData\Local\Programs\Python\Python311\Lib\site-packages\pand as\io\parsers\
 ↪py:1620, in TextFileReader.__init__(self, f, engine, **kwds)
   1617     self.options["has_index_names"] = kwds["has_index_names"]
   1619 self.handles: IOHandles | None = None
-> 1620 self._engine = self._make_engine(f, self.engine)

File c:
 ↪\Users\Prateek\AppData\Local\Programs\Python\Python311\Lib\site-packages\pand as\io\parsers\
 ↪py:1880, in TextFileReader._make_engine(self, f, engine)
   1878     if "b" not in mode:
   1879         mode += "b"
-> 1880 self.handles = get_handle(
   1881     f,
   1882     mode,
   1883     encoding=self.options.get("encoding", None),
   1884     compression=self.options.get("compression", None),
   1885     memory_map=self.options.get("memory_map", False),
   1886     is_text=is_text,
   1887     errors=self.options.get("encoding_errors", "strict"),
   1888     storage_options=self.options.get("storage_options", None),
   1889 )
   1890 assert self.handles is not None
   1891 f = self.handles.handle

File c:
 ↪\Users\Prateek\AppData\Local\Programs\Python\Python311\Lib\site-packages\pand as\io\common.
 ↪py:873, in get_handle(path_or_buf, mode, encoding, compression, memory_map,␣
 ↪is_text, errors, storage_options)
    868 elif isinstance(handle, str):
    869     # Check whether the filename is to be opened in binary mode.
    870     # Binary mode does not support 'encoding' and 'newline'.
    871     if ioargs.encoding and "b" not in ioargs.mode:
    872         # Encoding
--> 873         handle = open(
    874             handle,
    875             ioargs.mode,
    876             encoding=ioargs.encoding,
    877             errors=errors,
    878             newline="",
```

```
    879         )
    880     else:
    881         # Binary mode
    882         handle = open(handle, ioargs.mode)

FileNotFoundError: [Errno 2] No such file or directory: 'loan_data_set.csv'
```

[ ]:
```
to_drop = ['Gender','Married']
#df.drop(columns=to_drop, inplace=True)
df.drop(to_drop, inplace=True, axis=1)
```

[ ]:
```
df.head()
```

[ ]:
```
    Loan_ID Dependents     Education Self_Employed  ApplicantIncome  \
0  LP001002          0      Graduate            No             5849
1  LP001003          1      Graduate            No             4583
2  LP001005          0      Graduate           Yes             3000
3  LP001006          0  Not Graduate            No             2583
4  LP001008          0      Graduate            No             6000

   CoapplicantIncome  LoanAmount  Loan_Amount_Term  Credit_History  \
0                0.0         NaN             360.0             1.0
1             1508.0       128.0             360.0             1.0
2                0.0        66.0             360.0             1.0
3             2358.0       120.0             360.0             1.0
4                0.0       141.0             360.0             1.0

  Property_Area Loan_Status
0         Urban           Y
1         Rural           N
2         Urban           Y
3         Urban           Y
4         Urban           Y
```

[ ]:
```
df = pd.DataFrame({
    'brand': ['Yum Yum', 'Yum Yum', 'Indomie', 'Indomie', 'Indomie'],
    'style': ['cup', 'cup', 'cup', 'pack', 'pack'],
    'rating': [4, 4, 3.5, 15, 5]
})
df
```

[ ]:
```
     brand  rating style
0  Yum Yum     4.0   cup
1  Yum Yum     4.0   cup
2  Indomie     3.5   cup
3  Indomie    15.0  pack
4  Indomie     5.0  pack
```

[ ]:
```
df.drop_duplicates()
```

[ ]:

	brand	rating	style
0	Yum Yum	4.0	cup
2	Indomie	3.5	cup
3	Indomie	15.0	pack
4	Indomie	5.0	pack

[ ]:
```
#To remove duplicates on specific column(s), use subset.
df.drop_duplicates(subset=['brand'])
```

[ ]:

	brand	rating	style
0	Yum Yum	4.0	cup
2	Indomie	3.5	cup

[ ]:
```
#To remove duplicates on specific column(s), use subset.
#to remove duplicates and keep last occurrences, use keep.
df.drop_duplicates(subset=['brand', 'style'], keep='last')
```

[ ]:

	brand	rating	style
1	Yum Yum	4.0	cup
2	Indomie	3.5	cup
4	Indomie	5.0	pack

[ ]:
```
#https://pandas.pydata.org/docs/reference/frame.html
```