iris_analysis

August 12, 2025

0.0.1 task

load read save, visualize, preprocess dirty_iris dataset

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

[2]:
```
df=pd.read_csv("Iris.csv")
dirtyf=pd.read_csv("dirty_iris.csv")
```

[22]:
```
dirtyf.rename?
```

```
Signature:
dirtyf.rename(

    mapper: 'Renamer | None' = None,

    *,

    index: 'Renamer | None' = None,

    columns: 'Renamer | None' = None,

    axis: 'Axis | None' = None,

    copy: 'bool | None' = None,

    inplace: 'bool' = False,

    level: 'Level | None' = None,

    errors: 'IgnoreRaise' = 'ignore',

) -> 'DataFrame | None'
Docstring:
Rename columns or index labels.

Function / dict values must be unique (1-to-1). Labels not contained in
a dict / Series will be left as-is. Extra labels listed don't throw an
error.

See the :ref:`user guide <basics.rename>` for more.

Parameters
```

```
----------
mapper : dict-like or function
    Dict-like or function transformations to apply to
    that axis' values. Use either ``mapper`` and ``axis`` to
    specify the axis to target with ``mapper``, or ``index`` and
    ``columns``.
index : dict-like or function
    Alternative to specifying axis (``mapper, axis=0``
    is equivalent to ``index=mapper``).
columns : dict-like or function
    Alternative to specifying axis (``mapper, axis=1``
    is equivalent to ``columns=mapper``).
axis : {0 or 'index', 1 or 'columns'}, default 0
    Axis to target with ``mapper``. Can be either the axis name
    ('index', 'columns') or number (0, 1). The default is 'index'.
copy : bool, default True
    Also copy underlying data.

    .. note::
        The `copy` keyword will change behavior in pandas 3.0.
        `Copy-on-Write
        <https://pandas.pydata.org/docs/dev/user_guide/copy_on_write.html>`__
        will be enabled by default, which means that all methods with a
        `copy` keyword will use a lazy copy mechanism to defer the copy and
        ignore the `copy` keyword. The `copy` keyword will be removed in a
        future version of pandas.

        You can already get the future behavior and improvements through
        enabling copy on write ``pd.options.mode.copy_on_write = True``
inplace : bool, default False
    Whether to modify the DataFrame rather than creating a new one.
    If True then value of copy is ignored.
level : int or level name, default None
    In case of a MultiIndex, only rename labels in the specified
    level.
errors : {'ignore', 'raise'}, default 'ignore'
    If 'raise', raise a `KeyError` when a dict-like `mapper`, `index`,
    or `columns` contains labels that are not present in the Index
    being transformed.
    If 'ignore', existing keys will be renamed and extra keys will be
    ignored.

Returns
-------
DataFrame or None
    DataFrame with the renamed axis labels or None if ``inplace=True``.

Raises
```

```
------
KeyError
    If any of the labels is not found in the selected axis and
    "errors='raise'".

See Also
--------
DataFrame.rename_axis : Set the name of the axis.

Examples
--------
``DataFrame.rename`` supports two calling conventions

* ``(index=index_mapper, columns=columns_mapper, …)``
* ``(mapper, axis={'index', 'columns'}, …)``

We *highly* recommend using keyword arguments to clarify your
intent.

Rename columns using a mapping:

>>> df = pd.DataFrame({"A": [1, 2, 3], "B": [4, 5, 6]})
>>> df.rename(columns={"A": "a", "B": "c"})
   a  c
0  1  4
1  2  5
2  3  6

Rename index using a mapping:

>>> df.rename(index={0: "x", 1: "y", 2: "z"})
   A  B
x  1  4
y  2  5
z  3  6

Cast index labels to a different type:

>>> df.index
RangeIndex(start=0, stop=3, step=1)
>>> df.rename(index=str).index
Index(['0', '1', '2'], dtype='object')

>>> df.rename(columns={"A": "a", "B": "b", "C": "c"}, errors="raise")
Traceback (most recent call last):
KeyError: ['C'] not found in axis

Using axis-style parameters:
```

```
>>> df.rename(str.lower, axis='columns')
   a  b
0  1  4
1  2  5
2  3  6

>>> df.rename({1: 2, 2: 4}, axis='index')
   A  B
0  1  4
2  2  5
4  3  6
File:      c:\users\user\miniconda3\lib\site-packages\pandas\core\frame.py
Type:      method
```

showing first 5 rows and column names and renaming the dataset columns dirty_iris dataset

[ ]:
```
df.head()
dirtyf.head(10)
dirtyf.columns
# dirtyf.size
dirtyf.rename(columns={'sepal length (cm)':'sepal_length', 'sepal width (cm)':␣
↪'sepal_width', 'petal length (cm)':'petal_length', 'petal width (cm)':␣
↪'petal_width'},inplace=True)
```

[25]:
```
## coding
dirtyf
```

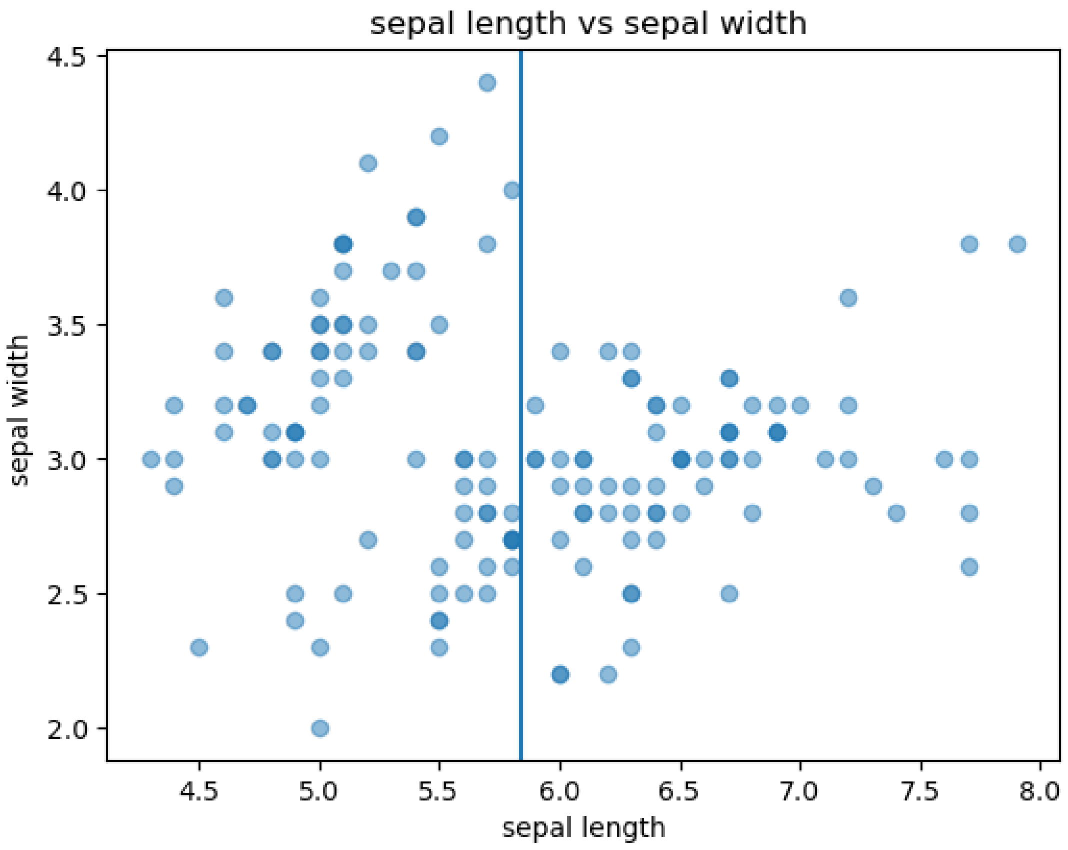
[25]:
```
     sepal_length  sepal_width  petal_length  petal_width    species
0             5.1          3.5           1.4          0.2     setosa
1             4.9          3.0           NaN          0.2     setosa
2             4.7          3.2           NaN          0.2     setosa
3             4.6          3.1           1.5          0.2     setosa
4             5.0          3.6           1.4          0.2     setosa
..            …            …             …            …          …
145           6.7          3.0           5.2          2.3  virginica
146           6.3          2.5           NaN          1.9  virginica
147           6.5          3.0           5.2          2.0  virginica
148           6.2          3.4           5.4          2.3  virginica
149           5.9          3.0           5.1          NaN  virginica

[150 rows x 5 columns]
```

[13]:
```
plt.axline?
```

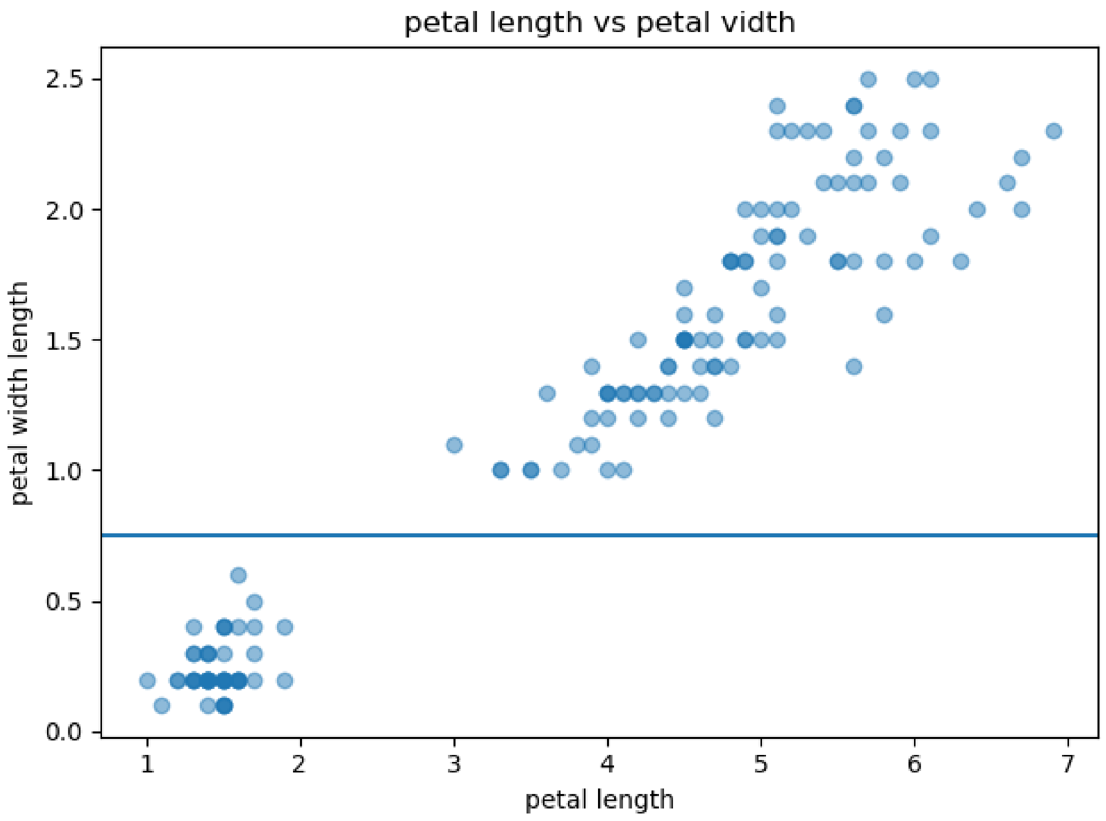
[ ]:
```
df['Species'].value_counts()
# dirtyf['species'].value_counts()
```

Datavisualization using scatter plot from matplotlib

[62]:
```
plt.scatter(x=df["SepalLengthCm"], y=df["SepalWidthCm"],alpha=0.5)
plt.title(' sepal length vs sepal width')
plt.axvline(5.843333333333334)
plt.xlabel('sepal length')
plt.ylabel('sepal width ')
plt.show()
```



[20]:
```
plt.scatter(x=df["PetalLengthCm"], y=df["PetalWidthCm"],alpha=0.5)
plt.title('petal length vs petal vidth')
plt.xlabel('petal length')
plt.ylabel('petal width length')
plt.axhline(0.75)
plt.tight_layout()
plt.show()
plt.savefig('petal_length_petal_width.png')
```



```
<Figure size 640x480 with 0 Axes>
```

Data-preprocessing

[16]:
```
dirtyf["species"].values
```

[16]:
```
array(['setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica'], dtype=object)
```

[31]:
```
dirtyf.dropna()
```

[31]:
```
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                  5.1               3.5                1.4               0.2
3                  4.6               3.1                1.5               0.2
4                  5.0               3.6                1.4               0.2
5                  5.4               3.9                1.7               0.4
6                  4.6               3.4                1.4               0.3
..                 …                …                 …                …
143                6.8               3.2                5.9               2.3
144                6.7               3.3                5.7               2.5
145                6.7               3.0                5.2               2.3
147                6.5               3.0                5.2               2.0
148                6.2               3.4                5.4               2.3

       species
0       setosa
3       setosa
4       setosa
5       setosa
6       setosa
..         …
143  virginica
144  virginica
145  virginica
147  virginica
148  virginica

[88 rows x 5 columns]
```

[36]:
```
dirtyf.isna()
```

```
[36]:      sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
      0                False             False              False             False
      1                False             False               True             False
      2                False             False               True             False
      3                False             False              False             False
      4                False             False              False             False
      ..                 …                 …                 …                …
      145              False             False              False             False
      146              False             False               True             False
      147              False             False              False             False
      148              False             False              False             False
      149              False             False              False              True

           species
      0      False
      1      False
      2      False
      3      False
      4      False
      ..       …
      145    False
      146    False
      147    False
      148    False
      149    False

      [150 rows x 5 columns]
```

```
[18]: dirtyf.describe()
```

```
[18]:        sepal length (cm)  sepal width (cm)  petal length (cm)  \
      count         127.000000        135.000000         132.000000
      mean            5.866929          3.049630           3.737879
      std             0.816599          0.445521           1.745421
      min             4.300000          2.000000           1.000000
      25%             5.100000          2.800000           1.600000
      50%             5.800000          3.000000           4.300000
      75%             6.400000          3.300000           5.100000
      max             7.900000          4.400000           6.900000

             petal width (cm)
      count        134.000000
      mean           1.159701
      std            0.768273
      min            0.100000
      25%            0.300000
      50%            1.300000
```

```
75%            1.800000
max            2.500000
```

[49]:
```
# dirtyf.replace('NaN', 4,inplace=True) # not working
dirtyf.fillna(10)
```

[49]:
```
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                  5.1               3.5                1.4               0.2
1                  4.9               3.0               10.0               0.2
2                  4.7               3.2               10.0               0.2
3                  4.6               3.1                1.5               0.2
4                  5.0               3.6                1.4               0.2
..                 …                 …                  …                 …
145                6.7               3.0                5.2               2.3
146                6.3               2.5               10.0               1.9
147                6.5               3.0                5.2               2.0
148                6.2               3.4                5.4               2.3
149                5.9               3.0                5.1              10.0

       species
0       setosa
1       setosa
2       setosa
3       setosa
4       setosa
..         …
145  virginica
146  virginica
147  virginica
148  virginica
149  virginica

[150 rows x 5 columns]
```

[53]:
```
df['SepalLengthCm'].mean()
```

[53]:
```
np.float64(5.843333333333334)
```

[54]:
```
df['SepalLengthCm'].median()
```

[54]:
```
np.float64(5.8)
```

Keyboard shortcuts:

- ctrl +o : shift between windows
- esc+m: markdown
- esc+c: code
- tab: new launcher

- ctrl+enter: run code
- shift +enter : run and move to new tab
- esc + a or b: new cell above or below

```
[ ]:
```