

---

# Python Data Audit



## **PyAudit: Python Data Audit Library API**

*Release 1.00*

**Wenqiang Feng and Ming Chen**

**May 01, 2019**



# CONTENTS

<b>1</b>	<b>Preface</b>	<b>3</b>
1.1	About . . . . .	3
1.1.1	About this API . . . . .	3
1.1.2	About the author . . . . .	3
1.2	Feedback and suggestions . . . . .	4
<b>2</b>	<b>How to Install</b>	<b>5</b>
2.1	Install with <code>pip</code> . . . . .	5
2.2	Install from Repo . . . . .	5
2.2.1	Clone the Repository . . . . .	5
2.2.2	Install . . . . .	5
2.2.3	Uninstall . . . . .	5
2.2.4	Test . . . . .	6
<b>3</b>	<b>Python Data Audit Functions</b>	<b>11</b>
3.1	Basic Functions . . . . .	11
3.1.1	<code>dtypes_class</code> . . . . .	11
3.1.2	<code>missing_rate</code> . . . . .	11
3.1.3	<code>zero_rate</code> . . . . .	12
3.1.4	<code>feature_variance</code> . . . . .	12
3.1.5	<code>freq_items_df</code> . . . . .	13
3.1.6	<code>feature_len</code> . . . . .	14
3.1.7	<code>correlation matrix</code> . . . . .	14
3.2	Summary Functions . . . . .	15
3.2.1	<code>numeric_summary</code> . . . . .	15
3.2.2	<code>category_summary</code> . . . . .	16
3.3	Auditing Function . . . . .	16
3.3.1	<code>auditing</code> . . . . .	16
<b>4</b>	<b>Auditing Demos</b>	<b>19</b>
4.1	Auditing in one function . . . . .	19
4.2	Auditing function by function . . . . .	21

<b>5 Main Reference</b>	<b>29</b>
<b>Bibliography</b>	<b>31</b>
<b>Python Module Index</b>	<b>33</b>
<b>Index</b>	<b>35</b>

# Python Data Audit



Welcome to our **PyAudit: Python Data Audit Library API**! The PDF version can be downloaded from [HERE](#).

You can install the PyAudit from [PyPI](<https://pypi.org/project/PyAudit>):

```
pip install PyAudit
```



## PREFACE

---

### Chinese proverb

Good tools are prerequisite to the successful execution of a job. – old Chinese proverb

---

## 1.1 About

### 1.1.1 About this API

This document is the API book for our PyAudit: Python Data Audit Library [PyAudit] API. The PDF version can be downloaded from [HERE](#). **You may download and distribute it. Please be aware, however, that the note contains typos as well as inaccurate or incorrect description.**

The API assumes that the reader has a preliminary knowledge of `python` programing and `Linux`. And this document is generated automatically by using `sphinx`.

### 1.1.2 About the author

- **Wenqiang Feng**
  - Sr. Data Scientist and PhD in Mathematics
  - University of Tennessee at Knoxville
  - Webpage: <http://web.utk.edu/~wfeng1/>
  - Email: [von198@gmail.com](mailto:von198@gmail.com)
- **Ming Chen**
  - Data Scientist and PhD in Genome Science and Technology
  - University of Tennessee at Knoxville

– Email: [ming.chen0919@gmail.com](mailto:ming.chen0919@gmail.com)

- **Biography**

Wenqiang Feng is Data Scientist within DST’s Applied Analytics Group. Dr. Feng’s responsibilities include providing DST clients with access to cutting-edge skills and technologies, including Big Data analytic solutions, advanced analytic and data enhancement techniques and modeling.

Dr. Feng has deep analytic expertise in data mining, analytic systems, machine learning algorithms, business intelligence, and applying Big Data tools to strategically solve industry problems in a cross-functional business. Before joining DST, Dr. Feng was an IMA Data Science Fellow at The Institute for Mathematics and its Applications (IMA) at the University of Minnesota. While there, he helped startup companies make marketing decisions based on deep predictive analytics.

Dr. Feng graduated from University of Tennessee, Knoxville, with Ph.D. in Computational Mathematics and Master’s degree in Statistics. He also holds Master’s degree in Computational Mathematics from Missouri University of Science and Technology (MST) and Master’s degree in Applied Mathematics from the University of Science and Technology of China (USTC).

- **Declaration**

The work of Wenqiang Feng was supported by the IMA, while working at IMA. However, any opinion, finding, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the IMA, UTK and DST.

## 1.2 Feedback and suggestions

Your comments and suggestions are highly appreciated. I am more than happy to receive corrections, suggestions or feedbacks through email (Wenqiang Feng: [von198@gmail.com](mailto:von198@gmail.com) and Ming Chen: [ming.chen0919@gmail.com](mailto:ming.chen0919@gmail.com)) for improvements.



## HOW TO INSTALL

### 2.1 Install with `pip`

You can install the `PyAudit` from [PyPI](<https://pypi.org/project/PyAudit>):

```
pip install PyAudit
```

### 2.2 Install from Repo

#### 2.2.1 Clone the Repository

```
git clone https://github.com/runawayhorse001/PyAudit.git
```

#### 2.2.2 Install

```
cd PyAudit
pip install -r requirements.txt
python setup.py install
```

#### 2.2.3 Uninstall

```
pip uninstall statspy
```

## 2.2.4 Test

```
cd PyAudit/test
python test.py
```

test.py

```
from PyAudit.basics import missing_rate, zero_rate, dtypes_class
from PyAudit.basics import feature_variance, freq_items_df, feature_len
from PyAudit.basics import corr_matrix, numeric_summary, category_
    ↳summary
import pandas as pd
import os, sys

output = os.path.abspath(os.path.join(sys.path[0])) + '/output'
print(output)
d = {'A': [1, 0, None, 3],
     'B': [1, 0, 0, 0],
     'C': ['a', None, 'c', 'd']}

# create DataFrame
df = pd.DataFrame(d)
print(missing_rate(df))
print(zero_rate(df))
print(feature_variance(df))
print(df)
print(feature_len(df))
print(numeric_summary(df, output))
print(category_summary(df, output))
print(corr_matrix(df, output))

d = {
    'num': list('1223334444'),
    'cat': list('wxyyyzzzz')
}
df = pd.DataFrame(d)
df = df.astype({"num": int, "cat": object})
print(freq_items_df(df, top_n=4))

# read df
df = pd.read_csv('Heart.csv', dtype={'Sex': bool})
print(df.head(5))
(num_fields, cat_fields, bool_fields, data_types, type_class) = dtypes_
    ↳class(df)
```

(continues on next page)

(continued from previous page)

```

print(num_fields)
print(cat_fields)
print(bool_fields)
print(data_types)
print(type_class)
print(missing_rate(df))
print(zero_rate(df))

print(freq_items_df(df, top_n=4))
print(feature_len(df))
print(numeric_summary(df, output))
print(category_summary(df, output))
print(corr_matrix(df, output))

```

Results:

```

feature  missing_rate
0        A           0.25
1        B           0.00
2        C           0.25

feature  zero_rate
0        A    0.333333
1        B    0.750000
2        C    0.000000

feature  feature_variance
0        A             1.0
1        B             0.5
2        C             1.0

  Age      Sex  ChestPain  RestBP  Chol  ...  Oldpeak  Slope  Ca
→  Thal  AHD
0   63   True    typical    145   233  ...    2.3     3  0.0
→  fixed   No
1   67   True  asymptomatic    160   286  ...    1.5     2  3.0
→  normal Yes
2   67   True  asymptomatic    120   229  ...    2.6     2  2.0
→reversible Yes
3   37   True   nonanginal    130   250  ...    3.5     3  0.0
→  normal   No
4   41  False  nontypical    130   204  ...    1.4     1  0.0
→  normal   No

[5 rows x 14 columns]
['Age', 'RestBP', 'Chol', 'Fbs', 'RestECG', 'MaxHR', 'ExAng', 'Oldpeak
→', 'Slope', 'Ca']
['ChestPain', 'Thal', 'AHD']

```

(continues on next page)

(continued from previous page)

```

['Sex']
      feature  dtypes
0      Age    int64
1      Sex     bool
2  ChestPain  object
3      RestBP  int64
4      Chol   int64
5      Fbs    int64
6  RestECG   int64
7      MaxHR  int64
8      ExAng  int64
9      Oldpeak float64
10     Slope   int64
11      Ca    float64
12     Thal   object
13     AHD    object

      feature  dtypes  class
0      Age    int64  numeric
1      Sex     bool   bool
2  ChestPain  object  category
3      RestBP  int64  numeric
4      Chol   int64  numeric
5      Fbs    int64  numeric
6  RestECG   int64  numeric
7      MaxHR  int64  numeric
8      ExAng  int64  numeric
9      Oldpeak float64  numeric
10     Slope   int64  numeric
11      Ca    float64  numeric
12     Thal   object  category
13     AHD    object  category

      feature  missing_rate
0      Age    0.000000
1      Sex    0.000000
2  ChestPain  0.000000
3      RestBP  0.000000
4      Chol   0.000000
5      Fbs    0.000000
6  RestECG   0.000000
7      MaxHR  0.000000
8      ExAng  0.000000
9      Oldpeak 0.000000
10     Slope   0.000000
11      Ca    0.013201
12     Thal   0.006601

```

(continues on next page)

(continued from previous page)

```
13          AHD          0.000000
```

```
Process finished with exit code 0
```



## PYTHON DATA AUDIT FUNCTIONS

### 3.1 Basic Functions

#### 3.1.1 dtypes\_class

`PyAudit.basics.dtypes_class(df_in)`  
numerical, categorical and bool name list in the DataFrame

**Parameters** `df_in` – input pandas DataFrame

**Returns** numerical, categorical and bool name list

**Author** Wenqiang Feng and Ming Chen

**Email** [von198@gmail.com](mailto:von198@gmail.com)

```
>>> from PyAudit.basics import dtypes_class
>>> df = pd.read_csv('Heart.csv', dtype={'Sex': bool})
>>> (num_fields, cat_fields, bool_fields, data_types, type_class),
    => dtypes_class(df)
>>> num_fields
['Age', 'RestBP', 'Chol', 'Fbs', 'RestECG', 'MaxHR', 'ExAng',
 => 'Oldpeak', 'Slope', 'Ca']
```

#### 3.1.2 missing\_rate

`PyAudit.basics.missing_rate(df_in)`  
calculate missing rate for each feature in the DataFrame

**Parameters** `df_in` – input pandas DataFrame

**Returns** missing rate

**Author** Wenqiang Feng and Ming Chen

**Email** [von198@gmail.com](mailto:von198@gmail.com)

```
>>> import pandas as pd
>>> d = {'A': [1, 0, None, 3],
        'B': [1, 0, 0, 0],
        'C': ['a', None, 'c', 'd']}
>>> # create DataFrame
>>> df = pd.DataFrame(d)
>>> from PyAudit.basics import missing_rate
>>> missing_rate(df)
      feature  missing_rate
0          A           0.25
1          B           0.00
2          C           0.25
```

### 3.1.3 zero\_rate

`PyAudit.basics.zero_rate(df_in)`

calculate the percentage of 0 value for each feature in the DataFrame

**Parameters** `df_in` – input pandas DataFrame

**Returns** zero rate

**Author** Wenqiang Feng and Ming Chen

**Email** von198@gmail.com

```
>>> import pandas as pd
>>> d = {'A': [1, 0, None, 3],
        'B': [1, 0, 0, 0],
        'C': ['a', None, 'c', 'd']}
>>> # create DataFrame
>>> df = pd.DataFrame(d)
>>> from PyAudit.basics import zero_rate
>>> zero_rate(df)
      feature  zero_rate
0          A   0.333333
1          B   0.750000
2          C   0.000000
```

### 3.1.4 feature\_variance

`PyAudit.basics.feature_variance(df_in)`

calculate the variance for each feature

**Parameters** `df_in` – input pandas DataFrame



**Returns** feature variance

**Author** Wenqiang Feng and Ming Chen

**Email** von198@gmail.com

```

>>> import pandas as pd
>>> d = {'A': [1, 0, None, 3],
        'B': [1, 0, 0, 0],
        'C': ['a', None, 'c', 'd']}
>>> # create DataFrame
>>> df = pd.DataFrame(d)
>>> from PyAudit.basics import zero_rate
>>> zero_rate(df)

```

	feature	feature_variance
0	A	1.0
1	B	0.5
2	C	1.0

### 3.1.5 freq\_items\_df

`PyAudit.basics.freq_items_df(df_in, top_n=3)`

find out the top n values and the corresponding frequency for each feature

**Parameters**

- **df\_in** – input pandas DataFrame
- **top\_n** – the number of the top values

**Returns** top n values and the corresponding frequency for each feature

**Author** Wenqiang Feng and Ming Chen

**Email** von198@gmail.com

```

>>> d = {
>>>     'num': list('1223334444'),
>>>     'cat': list('wxyyyzzzz')
>>> }
>>> df = pd.DataFrame(d)
>>> df = df.astype({"num": int, "cat": object})
>>> print(freq_items_df(df, top_n=4))

```

	feature	top_items	top_freqs
0	num	[4, 3, 2, 1]	[4, 3, 2, 1]
1	cat	[z, y, x, w]	[4, 3, 2, 1]

### 3.1.6 feature\_len

`PyAudit.basics.feature_len(df_in)`

find out the min and max length of values for each feature

**Parameters** `df_in` – input pandas DataFrame

**Returns** min and max length DataFrame

**Author** Wenqiang Feng and Ming Chen

**Email** [von198@gmail.com](mailto:von198@gmail.com)

```
>>> d = {'A': [1, 0, None, 3],
>>>        'B': [1, 0, 0, 0],
>>>        'C': ['a', None, 'c', 'd']}
>>> # create DataFrame
>>> df = pd.DataFrame(d)
>>> print(df)
   A  B  C
0  1  1  a
1  0  0 None
2 NaN 0  c
3  3  0  d
>>> print(feature_len(df))
   feature  min_length  max_length
0         A           3           3
1         B           1           1
2         C           1           4
```

### 3.1.7 correlation matrix

`PyAudit.basics.corr_matrix(df_in, output_dir)`

generate correlation matrix for numerical dataframe

**Parameters**

- `df_in` – input pandas DataFrame
- `output_dir` – output path

**Returns** correlation matrix

**Author** Wenqiang Feng and Ming Chen

**Email** [von198@gmail.com](mailto:von198@gmail.com)

```
>>> d = {'A': [1, 0, None, 3],
>>>        'B': [1, 0, 0, 0],
```

(continues on next page)

(continued from previous page)

```

>>>      'C': ['a', None, 'c', 'd']}
>>> # create DataFrame
>>> df = pd.DataFrame(d)
>>> print(corr_matrix(df))
           A          B
A  1.000000 -0.188982
B -0.188982  1.000000

```

## 3.2 Summary Functions

### 3.2.1 numeric\_summary

`PyAudit.basics.numeric_summary(df_in, output_dir, top_n=4, deciles=False)`  
 generate statistical summary for numerical DataFrame

#### Parameters

- **df\_in** – input pandas DataFrame
- **output\_dir** – output files directory
- **top\_n** – the number of the top item to show
- **deciles** – flag for percentiles style

**Returns** statistical summary for numerical data

**Author** Wenqiang Feng and Ming Chen

**Email** [von198@gmail.com](mailto:von198@gmail.com)

```

>>> d = {'A': [1, 0, None, 3],
>>>       'B': [1, 0, 0, 0],
>>>       'C': ['a', None, 'c', 'd']}
>>> # create DataFrame
>>> df = pd.DataFrame(d)
>>> print(numeric_summary(df))
  feature data_type  min_digits  ...  zero_rate  pos_rate  neg_
↪rate
  A          A    float64        3  ...    0.333333  0.666667  ↪
↪ 0.0
  B          B     int64        3  ...    0.750000  0.250000  ↪
↪ 0.0

```

## 3.2.2 category\_summary

`PyAudit.basics.category_summary(df_in, output_dir, top_n=4)`  
generate statistical summary for numerical DataFrame

### Parameters

- **df\_in** – input pandas DataFrame
- **output\_dir** – output files directory
- **top\_n** – the number of the top item to show

**Returns** statistical summary for numerical data

**Author** Wenqiang Feng and Ming Chen

**Email** [von198@gmail.com](mailto:von198@gmail.com)

```
>>> d = {'A': [1, 0, None, 3],
>>>       'B': [1, 0, 0, 0],
>>>       'C': ['a', None, 'c', 'd']}
>>> # create DataFrame
>>> df = pd.DataFrame(d)
>>> print(numeric_summary(df))
  feature data_type  min_digits  ...  top_values  top_freqs  _
->missing_rate
   C      C      object        1  ...    [a, d, c]  [1, 1, 1]  _
->      0.25
```

## 3.3 Auditing Function

### 3.3.1 auditing

`PyAudit.basics.auditing(df_in, output_dir, top_n=4, deciles=False)`  
generate audited results

### Parameters

- **df\_in** – input pandas DataFrame
- **output\_dir** – output files directory
- **top\_n** – the number of the top item to show
- **deciles** – flag for percentiles style

**Author** Wenqiang Feng and Ming Chen

**Email** [von198@gmail.com](mailto:von198@gmail.com)

```
>>> d = {'A': [1, 0, None, 3],
>>>       'B': [1, 0, 0, 0],
>>>       'C': ['a', None, 'c', 'd']}
>>> # create DataFrame
>>> df = pd.DataFrame(d)
>>> print(auditing(df,path))
```

	feature	data_type	min_digits	...	zero_rate	pos_rate	neg_
↪ rate							
	A	A	float64	3 ...	0.333333	0.666667	↪
↪ 0.0							
	B	B	int64	3 ...	0.750000	0.250000	↪
↪ 0.0							



## AUDITING DEMOS

The following demos are designed to show how to use `PyAudit` to audit `pd.DataFrame`.

### 4.1 Auditing in one function

For example:

```
# import python libraries
import os
import sys
import pandas as pd

# import PyAudit module
from PyAudit.basics import auditing

# Audit output path
output = os.path.abspath(os.path.join(sys.path[0])) + '/output'

# load DataFrame
df = pd.read_csv('Heart.csv', dtype={'Sex': bool})
print(df.head(5))

# generate the audit results (.csv files in output folder)
num_summary, cat_summary, corr = auditing(df, output)

# the following is optional, since the .csv files are in the output_
  ↳ folder
print(num_summary)
print(cat_summary)
print(corr)
```

Result:

	Age	Sex	ChestPain	RestBP	Chol	...	Oldpeak	Slope	Ca	└
→	Thal	AHD								
0	63	True	typical	145	233	...	2.3	3	0.0	└
→	fixed	No								
1	67	True	asymptomatic	160	286	...	1.5	2	3.0	└
→	normal	Yes								
2	67	True	asymptomatic	120	229	...	2.6	2	2.0	└
→	reversable	Yes								
3	37	True	nonanginal	130	250	...	3.5	3	0.0	└
→	normal	No								
4	41	False	nontypical	130	204	...	1.4	1	0.0	└
→	normal	No								
[5 rows x 14 columns]										
	feature	data_type	min_digits	...	zero_rate	pos_rate	neg_			
→	rate									
Age	Age	int64	4	...	0.000000	1.000000				└
→	0.0									
RestBP	RestBP	int64	4	...	0.000000	1.000000				└
→	0.0									
Chol	Chol	int64	5	...	0.000000	1.000000				└
→	0.0									
Fbs	Fbs	int64	3	...	0.851485	0.148515				└
→	0.0									
RestECG	RestECG	int64	3	...	0.498350	0.501650				└
→	0.0									
MaxHR	MaxHR	int64	4	...	0.000000	1.000000				└
→	0.0									
ExAng	ExAng	int64	3	...	0.673267	0.326733				└
→	0.0									
Oldpeak	Oldpeak	float64	3	...	0.326733	0.673267				└
→	0.0									
Slope	Slope	int64	3	...	0.000000	1.000000				└
→	0.0									
Ca	Ca	float64	3	...	0.588629	0.411371				└
→	0.0									
[10 rows x 21 columns]										
	feature	data_type	...	top_freqs	missing_rate					
Sex	Sex	bool	...	[206, 97]	0.000000					
ChestPain	ChestPain	object	...	[144, 86, 50, 23]	0.000000					
Thal	Thal	object	...	[166, 117, 18]	0.006601					
AHD	AHD	object	...	[164, 139]	0.000000					
[4 rows x 10 columns]										
	Age	RestBP	Chol	...	Oldpeak	Slope				└
→	Ca									

(continues on next page)



(continued from previous page)

```

Age      1.000000  0.284946  0.208950  ...  0.203805  0.161770  0.
↪362605
RestBP   0.284946  1.000000  0.130120  ...  0.189171  0.117382  0.
↪098773
Chol     0.208950  0.130120  1.000000  ...  0.046564 -0.004062  0.
↪119000
Fbs      0.118530  0.175340  0.009841  ...  0.005747  0.059894  0.
↪145478
RestECG  0.148868  0.146560  0.171043  ...  0.114133  0.133946  0.
↪128343
MaxHR    -0.393806 -0.045351 -0.003432  ... -0.343085 -0.385601 -0.
↪264246
ExAng    0.091661  0.064762  0.061310  ...  0.288223  0.257748  0.
↪145570
Oldpeak  0.203805  0.189171  0.046564  ...  1.000000  0.577537  0.
↪295832
Slope    0.161770  0.117382 -0.004062  ...  0.577537  1.000000  0.
↪110119
Ca       0.362605  0.098773  0.119000  ...  0.295832  0.110119  1.
↪000000

```

```
[10 rows x 10 columns]
```

```
Process finished with exit code 0
```

and

## 4.2 Auditing function by function

For example:

```

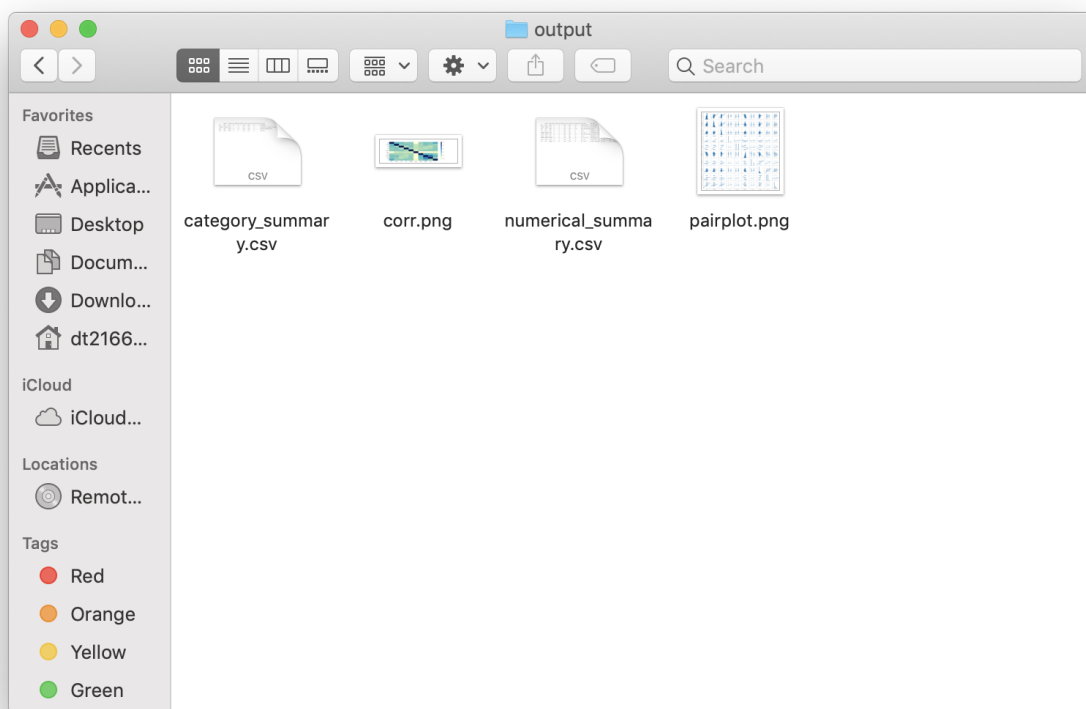
# import python libraries
import os
import sys
import pandas as pd

# import PyAudit module
from PyAudit.basics import corr_matrix, numeric_summary, category_
↪summary

# Audit output path
output = os.path.abspath(os.path.join(sys.path[0])) + '/output'

```

(continues on next page)



(continued from previous page)

```
# load DataFrame
df = pd.read_csv('Heart.csv', dtype={'Sex': bool})
print(df.head(5))

# generate the audit results (.csv files in output folder)
print(numeric_summary(df, output))
print(category_summary(df, output))
print(corr_matrix(df, output))
```

Result:

	Age	Sex	ChestPain	RestBP	Chol	...	Oldpeak	Slope	Ca	
→	Thal	AHD								
0	63	True	typical	145	233	...	2.3	3	0.0	
→	fixed	No								
1	67	True	asymptomatic	160	286	...	1.5	2	3.0	
→	normal	Yes								
2	67	True	asymptomatic	120	229	...	2.6	2	2.0	
→	reversible	Yes								
3	37	True	nonanginal	130	250	...	3.5	3	0.0	
→	normal	No								
4	41	False	nontypical	130	204	...	1.4	1	0.0	
→	normal	No								

[5 rows x 14 columns]

	feature	data_type	min_digits	...	zero_rate	pos_rate	neg_
→	rate						
Age	Age	int64	4	...	0.000000	1.000000	
→	0.0						
RestBP	RestBP	int64	4	...	0.000000	1.000000	
→	0.0						
Chol	Chol	int64	5	...	0.000000	1.000000	
→	0.0						
Fbs	Fbs	int64	3	...	0.851485	0.148515	
→	0.0						
RestECG	RestECG	int64	3	...	0.498350	0.501650	
→	0.0						
MaxHR	MaxHR	int64	4	...	0.000000	1.000000	
→	0.0						
ExAng	ExAng	int64	3	...	0.673267	0.326733	
→	0.0						
Oldpeak	Oldpeak	float64	3	...	0.326733	0.673267	
→	0.0						
Slope	Slope	int64	3	...	0.000000	1.000000	
→	0.0						

(continues on next page)

(continued from previous page)

```
Ca          Ca    float64          3 ...    0.588629  0.411371
→0.0

[10 rows x 21 columns]
           feature data_type ...           top_freqs  missing_rate
Sex          Sex      bool ...           [206, 97]          0.000000
ChestPain    ChestPain  object ...  [144, 86, 50, 23]          0.000000
Thal          Thal    object ...           [166, 117, 18]          0.006601
AHD           AHD     object ...           [164, 139]          0.000000

[4 rows x 10 columns]
           Age    RestBP    Chol ...    Oldpeak    Slope
→Ca
Age          1.000000  0.284946  0.208950 ...    0.203805  0.161770  0.
→362605
RestBP       0.284946  1.000000  0.130120 ...    0.189171  0.117382  0.
→098773
Chol         0.208950  0.130120  1.000000 ...    0.046564 -0.004062  0.
→119000
Fbs          0.118530  0.175340  0.009841 ...    0.005747  0.059894  0.
→145478
RestECG      0.148868  0.146560  0.171043 ...    0.114133  0.133946  0.
→128343
MaxHR       -0.393806 -0.045351 -0.003432 ...   -0.343085 -0.385601 -0.
→264246
ExAng        0.091661  0.064762  0.061310 ...    0.288223  0.257748  0.
→145570
Oldpeak      0.203805  0.189171  0.046564 ...    1.000000  0.577537  0.
→295832
Slope        0.161770  0.117382 -0.004062 ...    0.577537  1.000000  0.
→110119
Ca           0.362605  0.098773  0.119000 ...    0.295832  0.110119  1.
→000000

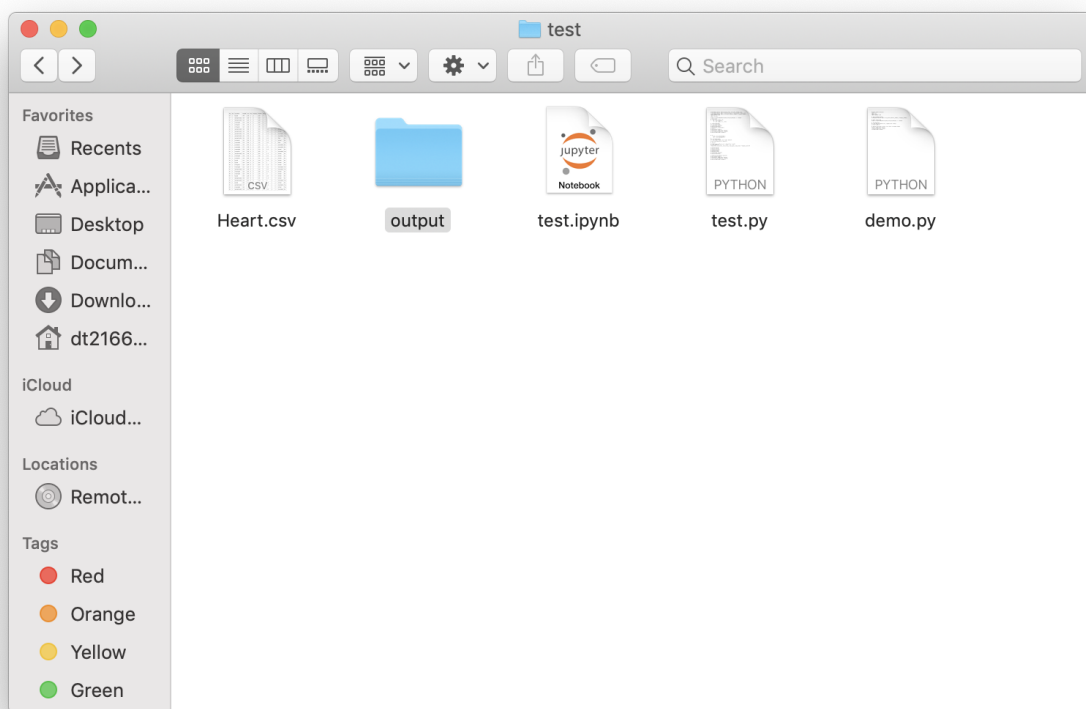
[10 rows x 10 columns]

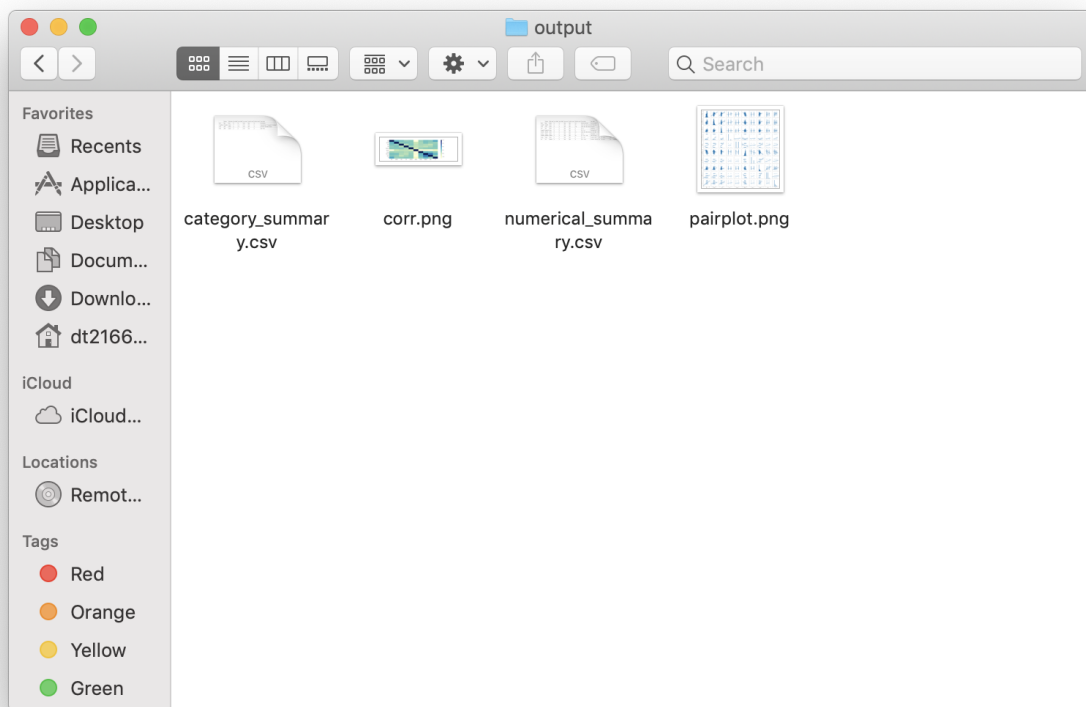
Process finished with exit code 0
```

and

```
....
iii*iiii
.-'`;-');;.
/' .-./ *;;
```

(continues on next page)





(continued from previous page)

```
. ' \d \;; .iii,  
/ o \; ,_. ,i*iii*,  
\_ , _._, ' \_.'') _)--.iiii;*iiii,  
\"\";iii\ /-'')_ _)\ \"' 'iiiiii;  
;*iii; -') `)_ _)\ | \ | ;iii*;  
iiii| \---\ o | | ;iii*;  
*;*;\| o / ;iii*;  
iiii;/| .-----\ / ;iii*;  
iii*;/ \ | ' ( ' . ;iii*;  
iiii;' . ; | ) \ | ;iii*;  
,*iiii;\| | . / / \ | 'iii*;  
iiii;/ | / / _/ 'iii;  
'*wF*/ | / _/ | ;*;  
\"\"\"\"\" \"\"\"\"\" ;'
```





**MAIN REFERENCE**



## BIBLIOGRAPHY

[PyAudit] Wenqiang Feng and Ming Chen. [Python Data Audit Library API](#), 2019.



## PYTHON MODULE INDEX

### p

`PyAudit.basics`, [11](#)



## INDEX

### D

`dtypes_class()` (*in module PyAudit basics*), 11

### P

`PyAudit.basics` (*module*), 11