Missing Data Duplicates Outliers sample dummies

In [95]:
```
import numpy as np
import pandas as pd
```

Handling Missing Data

In [96]:
```
from numpy import nan as NA
```

In [97]:
```
s = pd.Series([3, NA, 8, NA, 5])
s
```

Out[97]:
```
0    3.0
1    NaN
2    8.0
3    NaN
4    5.0
dtype: float64
```

In [98]:
```
s.notnull()
```

Out[98]:
```
0     True
1    False
2     True
3    False
4     True
dtype: bool
```

In [99]:
```
myser = s[s.notnull()]
myser
```

Out[99]:
```
0    3.0
2    8.0
4    5.0
dtype: float64
```

In [100]:
```
s.dropna()
```

Out[100]:
```
0    3.0
2    8.0
4    5.0
dtype: float64
```

In [101]:
```
#
```

In [102]:
```
df = pd.DataFrame([[4, 9, 5], [2, NA, NA], [NA, NA, NA], [NA, 8, 6]])
df
```

Out[102]:

	0	1	2
0	4.0	9.0	5.0
1	2.0	NaN	NaN
2	NaN	NaN	NaN
3	NaN	8.0	6.0

In [103]:
```
cleaned = df.dropna()
cleaned
```

Out[103]:

	0	1	2
0	4.0	9.0	5.0

In [104]:
```
df
```

Out[104]:

	0	1	2
0	4.0	9.0	5.0
1	2.0	NaN	NaN
2	NaN	NaN	NaN
3	NaN	8.0	6.0

In [105]:
```
df.dropna(how='all')
```

Out[105]:

	0	1	2
0	4.0	9.0	5.0
1	2.0	NaN	NaN
3	NaN	8.0	6.0

In [106]:
```
df
```

Out[106]:

	0	1	2
0	4.0	9.0	5.0
1	2.0	NaN	NaN
2	NaN	NaN	NaN
3	NaN	8.0	6.0

In [107]:
```
df[4] = NA
df
```

Out[107]:

	0	1	2	4
0	4.0	9.0	5.0	NaN
1	2.0	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN
3	NaN	8.0	6.0	NaN

In [108]:
```
df.dropna(axis=1, how='all')
```

Out[108]:

	0	1	2
0	4.0	9.0	5.0
1	2.0	NaN	NaN
2	NaN	NaN	NaN
3	NaN	8.0	6.0

In [109]:
```
#
```

In [110]:

```
d = [[14, NA, NA], [12, NA, NA], [18, NA, 7], [18, NA, 16], [15, 12, 19], [13
df = pd.DataFrame(d)
df
```

Out[110]:

	0	1	2
0	14	NaN	NaN
1	12	NaN	NaN
2	18	NaN	7.0
3	18	NaN	16.0
4	15	12.0	19.0
5	13	1.0	6.0

In [111]:

```
df.dropna()
```

Out[111]:

	0	1	2
4	15	12.0	19.0
5	13	1.0	6.0

In [112]:

```
df.dropna(thresh=2)
```

Out[112]:

	0	1	2
2	18	NaN	7.0
3	18	NaN	16.0
4	15	12.0	19.0
5	13	1.0	6.0

fillna : Filling In Missing Data

In [113]:

```
df
```

Out[113]:

	0	1	2
0	14	NaN	NaN
1	12	NaN	NaN
2	18	NaN	7.0
3	18	NaN	16.0
4	15	12.0	19.0
5	13	1.0	6.0

In [114]:
```
frame = df.fillna(0)
frame
```

Out[114]:

	0	1	2
0	14	0.0	0.0
1	12	0.0	0.0
2	18	0.0	7.0
3	18	0.0	16.0
4	15	12.0	19.0
5	13	1.0	6.0

In [115]:
```
df
```

Out[115]:

	0	1	2
0	14	NaN	NaN
1	12	NaN	NaN
2	18	NaN	7.0
3	18	NaN	16.0
4	15	12.0	19.0
5	13	1.0	6.0

In [116]:
```
df.fillna({1: 0, 2: 9})
```

Out[116]:

	0	1	2
0	14	0.0	9.0
1	12	0.0	9.0
2	18	0.0	7.0
3	18	0.0	16.0
4	15	12.0	19.0
5	13	1.0	6.0

In [117]:
```
df
```

Out[117]:

	0	1	2
0	14	NaN	NaN
1	12	NaN	NaN
2	18	NaN	7.0
3	18	NaN	16.0
4	15	12.0	19.0
5	13	1.0	6.0

In [118]:
```
_ = df.fillna(0, inplace=True)
```

In [119]:
```
df
```

Out[119]:

	0	1	2
0	14	0.0	0.0
1	12	0.0	0.0
2	18	0.0	7.0
3	18	0.0	16.0
4	15	12.0	19.0
5	13	1.0	6.0

In [120]:
```
#
```

In [121]:
```
d = [[15, 12, 19], [13, 1, 6], [18, NA, 7], [18, NA, 16], [14, NA, NA], [12,
df = pd.DataFrame(d)
df
```

Out[121]:

	0	1	2
0	15	12.0	19.0
1	13	1.0	6.0
2	18	NaN	7.0
3	18	NaN	16.0
4	14	NaN	NaN
5	12	NaN	NaN

In [122]:
```
df.fillna(method='ffill')
```

Out[122]:

	0	1	2
0	15	12.0	19.0
1	13	1.0	6.0
2	18	1.0	7.0
3	18	1.0	16.0
4	14	1.0	16.0
5	12	1.0	16.0

In [123]:
```
df.fillna(method='ffill', limit=3)
```

Out[123]:

	0	1	2
0	15	12.0	19.0
1	13	1.0	6.0
2	18	1.0	7.0
3	18	1.0	16.0
4	14	1.0	16.0
5	12	NaN	16.0

In [124]:
```
#
```

In [125]:
```
s = pd.Series([6, NA, 4, NA, 5])
s
```

Out[125]:
```
0    6.0
1    NaN
2    4.0
3    NaN
4    5.0
dtype: float64
```

In [126]:
```
s.mean()
```

Out[126]:
```
5.0
```

In [127]:
```
s.fillna(s.mean())
```

Out[127]:
```
0    6.0
1    5.0
2    4.0
3    5.0
4    5.0
dtype: float64
```

drop_duplicates : Removing Duplicates

In [128]:
```
frame = pd.DataFrame({'col1': ['a', 'b', 'a', 'b', 'a', 'b', 'b'],
                      'col2': [1, 1, 2, 3, 3, 4, 4]})
frame
```

Out[128]:

	col1	col2
0	a	1
1	b	1
2	a	2
3	b	3
4	a	3
5	b	4
6	b	4

In [129]:
```
frame.duplicated()
```

Out[129]:
```
0    False
1    False
2    False
3    False
4    False
5    False
6     True
dtype: bool
```

In [130]:
```
df = frame.drop_duplicates()
df
```

Out[130]:

	col1	col2
0	a	1
1	b	1
2	a	2
3	b	3
4	a	3
5	b	4

In [131]:
```
frame
```

Out[131]:

	col1	col2
0	a	1
1	b	1
2	a	2
3	b	3
4	a	3
5	b	4
6	b	4

In [132]:
```
frame.drop_duplicates(['col1'])
```

Out[132]:

	col1	col2
0	a	1
1	b	1

In [133]:
```
frame.drop_duplicates(['col1'], keep='last')
```

Out[133]:

	col1	col2
4	a	3
6	b	4

In [134]:
```
frame
```

Out[134]:

	col1	col2
0	a	1
1	b	1
2	a	2
3	b	3
4	a	3
5	b	4
6	b	4

In [135]:
```
frame.drop_duplicates(['col2'])
```

Out[135]:

	col1	col2
0	a	1
2	a	2
3	b	3
5	b	4

In [136]:
```
frame.drop_duplicates(['col2'], keep='last')
```

Out[136]:

	col1	col2
1	b	1
2	a	2
4	a	3
6	b	4

replace : Replacing Values

In [137]:
```
myser = pd.Series([13, 9, 20, 9, 18, 17])
myser
```

Out[137]:
```
0    13
1     9
2    20
3     9
4    18
5    17
dtype: int64
```

In [138]:
```
myser.replace(9, np.nan)
```

Out[138]:
```
0    13.0
1     NaN
2    20.0
3     NaN
4    18.0
5    17.0
dtype: float64
```

In [139]:
```
myser.replace({9: np.nan, 18: 0})
```

Out[139]:
```
0    13.0
1     NaN
2    20.0
3     NaN
4     0.0
5    17.0
dtype: float64
```

In [140]:
```
myser.replace([9, 18], [np.nan, 0])
```

Out[140]:
```
0    13.0
1     NaN
2    20.0
3     NaN
4     0.0
5    17.0
dtype: float64
```

cut

In [141]:
```
score = [20, 8, 17, 10]
bins  = [0, 9, 15, 20]
```

In [142]:
```
c = pd.cut(score, bins)
c.categories
```

Out[142]:
```
IntervalIndex([(0, 9], (9, 15], (15, 20]],
              closed='right',
              dtype='interval[int64]')
```

In [143]:
```
c
```

Out[143]:
```
[(15, 20], (0, 9], (15, 20], (9, 15]]
Categories (3, interval[int64]): [(0, 9] < (9, 15] < (15, 20]]
```

In [144]:
```
c.codes
```

Out[144]:
```
array([2, 0, 2, 1], dtype=int8)
```

In [145]:
```
pd.value_counts(c)
```

Out[145]:
```
(15, 20]    2
(0, 9]      1
(9, 15]     1
dtype: int64
```

In [146]:
```
#
```

In [147]:
```
score = [20, 8, 17, 10]
bins  = [0, 9, 15, 20]

x = pd.cut(score, bins, labels=["bad", "medium", "good" ])
x
```

Out[147]:
```
['good', 'bad', 'good', 'medium']
Categories (3, object): ['bad' < 'medium' < 'good']
```

In [148]:
```
pd.value_counts(x)
```

Out[148]:
```
good      2
bad       1
medium    1
dtype: int64
```

In [149]:
```
#
```

In [150]:
```
ages = [20, 58, 24, 72, 100]
bins = [18, 25, 35, 60, 110]
x = pd.cut(ages, bins)
pd.value_counts(x)
```

Out[150]:
```
(18, 25]     2
(60, 110]    2
(35, 60]     1
(25, 35]     0
dtype: int64
```

In [151]:
```
c= pd.cut(ages, bins, labels=['Youth', 'YoungAdult', 'MiddleAged', 'Senior'])
pd.value_counts(c)
```

Out[151]:
```
Youth         2
Senior        2
MiddleAged    1
YoungAdult    0
dtype: int64
```

Detecting Outliers

In [152]:
```
df = pd.DataFrame(np.random.randn(1000, 3))
df.describe()
```

Out[152]:

	0	1	2
count	1000.000000	1000.000000	1000.000000
mean	-0.026937	0.007819	0.017632
std	0.997187	1.024177	0.999954
min	-3.515552	-3.847616	-3.711631
25%	-0.696841	-0.701809	-0.623527
50%	-0.016783	0.013533	0.017140
75%	0.654072	0.717659	0.700989
max	3.819215	3.778746	3.411838

In [153]:
```
df.head()
```

Out[153]:

	0	1	2
0	0.896824	-1.900415	-0.019045
1	0.600042	-0.959378	-0.378446
2	0.109422	-0.954451	1.192668
3	-1.469650	-0.743459	0.699459
4	1.435612	-0.954270	0.291966

In [154]:
```
s = np.sign(df)
s.head()
```

Out[154]:

	0	1	2
0	1.0	-1.0	-1.0
1	1.0	-1.0	-1.0
2	1.0	-1.0	1.0
3	-1.0	-1.0	1.0
4	1.0	-1.0	1.0

In [155]:
```
df[np.abs(df) > 3] = s * 3
```

In [156]:
```
df.describe()
```

Out[156]:

	0	1	2
count	1000.000000	1000.000000	1000.000000
mean	-0.027111	0.007681	0.018221
std	0.992283	1.016044	0.994879
min	-3.000000	-3.000000	-3.000000
25%	-0.696841	-0.701809	-0.623527
50%	-0.016783	0.013533	0.017140
75%	0.654072	0.717659	0.700989
max	3.000000	3.000000	3.000000

take

In [157]:
```
df = pd.DataFrame(np.arange(12).reshape((4, 3)))
df
```

Out[157]:

	0	1	2
0	0	1	2
1	3	4	5
2	6	7	8
3	9	10	11

In [158]:
```
p = np.random.permutation(4)
p
```

Out[158]:
```
array([3, 0, 2, 1])
```

In [159]:
```
df.take(p)  # Return the elements in the given positional indices along an ax
```

Out[159]:

	0	1	2
3	9	10	11
0	0	1	2
2	6	7	8
1	3	4	5

sample : return a random sample of items from an axis of object.

In [160]:
```
df = pd.DataFrame(np.arange(30).reshape((6, 5)))
df
```

Out[160]:

	0	1	2	3	4
0	0	1	2	3	4
1	5	6	7	8	9
2	10	11	12	13	14
3	15	16	17	18	19
4	20	21	22	23	24
5	25	26	27	28	29

In [161]:
```python
df.sample(n=2)
```

Out[161]:

	0	1	2	3	4
1	5	6	7	8	9
4	20	21	22	23	24

In [162]:
```python
s = pd.Series([15, 7, 12, 16])
s.sample(n=2)
```

Out[162]:
```
3    16
1     7
dtype: int64
```

In [163]:
```python
#
```

In [164]:
```python
a = ['s1', 's2', 's3', 's4']
b = {'F1': [7, 4, 6, 0], 'F2': [2, 1, 3, 0],'F3': [9, 4, 1, 7]}
df = pd.DataFrame(data=b, index=a)
df
```

Out[164]:

	F1	F2	F3
s1	7	2	9
s2	4	1	4
s3	6	3	1
s4	0	0	7

In [165]:
```python
df.sample(n=2, random_state=1)
```

Out[165]:

	F1	F2	F3
s4	0	0	7
s3	6	3	1

In [166]:
```python
df.sample(n=2, weights='F3', random_state=1)
```

Out[166]:

	F1	F2	F3
s1	7	2	9
s4	0	0	7

In [167]:
```
s
```

Out[167]:
```
0    15
1     7
2    12
3    16
dtype: int64
```

In [168]:
```
s.sample(n=6, replace=True)
```

Out[168]:
```
0    15
0    15
0    15
2    12
2    12
1     7
dtype: int64
```

In [169]:
```
df
```

Out[169]:

	F1	F2	F3
s1	7	2	9
s2	4	1	4
s3	6	3	1
s4	0	0	7

In [170]:
```
df.sample(frac=0.5)
```

Out[170]:

	F1	F2	F3
s1	7	2	9
s2	4	1	4

In [171]:
```
df.sample(frac=2, replace=True)
```

Out[171]:

	F1	F2	F3
s4	0	0	7
s1	7	2	9
s4	0	0	7
s1	7	2	9
s2	4	1	4
s2	4	1	4
s2	4	1	4
s4	0	0	7

In [172]:
```
df
```

Out[172]:

	F1	F2	F3
s1	7	2	9
s2	4	1	4
s3	6	3	1
s4	0	0	7

In [173]:
```
myser = df['F1'].sample(n=3)
myser
```

Out[173]:
```
s2    4
s3    6
s1    7
Name: F1, dtype: int64
```

get_dummies

In [174]:
```
s = pd.Series(list('abca'))
s
```

Out[174]:
```
0    a
1    b
2    c
3    a
dtype: object
```

In [175]:
```
df = pd.get_dummies(s)
df
```

Out[175]:

	a	b	c
0	1	0	0
1	0	1	0
2	0	0	1
3	1	0	0

In [176]:
```
myser = pd.Series(list('abcaa'))
pd.get_dummies(myser)
```

Out[176]:

	a	b	c
0	1	0	0
1	0	1	0
2	0	0	1
3	1	0	0
4	1	0	0

In [177]:
```
lst = ['a', 'b', np.nan]
pd.get_dummies(lst)
```

Out[177]:

	a	b
0	1	0
1	0	1
2	0	0

In [178]:
```
#
```

In [179]:
```
df = pd.DataFrame({'F1': ['a', 'b', 'c'], 'F2': ['b', 'a', 'a'], 'F3': [7, 2,
df
```

Out[179]:

	F1	F2	F3
0	a	b	7
1	b	a	2
2	c	a	3

In [180]:
```
type(df['F1'])
```

Out[180]: pandas.core.series.Series

In [181]:
```
pd.get_dummies(df['F1'])
```

Out[181]:

	a	b	c
0	1	0	0
1	0	1	0
2	0	0	1

In [182]:
```
pd.get_dummies(df['F2'])
```

Out[182]:

	a	b
0	0	1
1	1	0
2	1	0

In [183]:
```
df
```

Out[183]:

	F1	F2	F3
0	a	b	7
1	b	a	2
2	c	a	3

In [184]:
```
pd.get_dummies(df)
```

Out[184]:

	F3	F1_a	F1_b	F1_c	F2_a	F2_b
0	7	1	0	0	0	1
1	2	0	1	0	1	0
2	3	0	0	1	1	0

In [185]:
```
pd.get_dummies(df, prefix=['col1', 'col2'])
```

Out[185]:

	F3	col1_a	col1_b	col1_c	col2_a	col2_b
0	7	1	0	0	0	1
1	2	0	1	0	1	0
2	3	0	0	1	1	0

In [186]:
```
d = pd.get_dummies(df['F1'])
d
```

Out[186]:

	a	b	c
0	1	0	0
1	0	1	0
2	0	0	1

In [187]:
```
df
```

Out[187]:

	F1	F2	F3
0	a	b	7
1	b	a	2
2	c	a	3

In [188]:
```
dfdummy = df[['F2','F3']].join(d)
dfdummy
```

Out[188]:

	F2	F3	a	b	c
0	b	7	1	0	0
1	a	2	0	1	0
2	a	3	0	0	1

دانشگاه شهید مدنی آذربایجان
برنامه نویسی پیشرفته با پایتون
امین گلزاری اسکوئی
۱۴۰۰-۱۴۰۱

[Codes and Projects (click here) (https://github.com/Amin-Golzari-Oskouei/Python-Programming-Course-Advanced-2021)](https://github.com/Amin-Golzari-Oskouei/Python-Programming-Course-Advanced-2021) [slides and videos (click here) (https://drive.google.com/drive/folders/1Dx3v7fD1QBWL-MNP2hd7iIxaRbeALkkA)](https://drive.google.com/drive/folders/1Dx3v7fD1QBWL-MNP2hd7iIxaRbeALkkA)