**SAMPLE CODINGS**

import pandas as pd

import matplotlib.pyplot as plt

from functools import reduce

df = pd.read\_csv('instagram.csv',encoding= 'unicode\_escape')

df

profile pic nums/length username fullname words nums/length fullname name==username description length external URL private #posts #followers #follows fake

0 1 0.27 0 0.00 0 53 0 0 32 1000 955 0

1 1 0.00 2 0.00 0 44 0 0 286 2740 533 0

2 1 0.10 2 0.00 0 0 0 1 13 159 98 0

3 1 0.00 1 0.00 0 82 0 0 679 414 651 0

4 1 0.00 2 0.00 0 0 0 1 6 151 126 0

... ... ... ... ... ... ... ... ... ... ... ... ...

571 1 0.55 1 0.44 0 0 0 0 33 166 596 1

572 1 0.38 1 0.33 0 21 0 0 44 66 75 1

573 1 0.57 2 0.00 0 0 0 0 4 96 339 1

574 1 0.57 1 0.00 0 11 0 0 0 57 73 1

575 1 0.27 1 0.00 0 0 0 0 2 150 487 1

576 rows × 12 columns

df.dtypes

profile pic int64

nums/length username float64

fullname words int64

nums/length fullname float64

name==username int64

description length int64

external URL int64

private int64

#posts int64

#followers int64

#follows int64

fake int64

dtype: object

df.isnull().sum()

profile pic 0

nums/length username 0

fullname words 0

nums/length fullname 0

name==username 0

description length 0

external URL 0

private 0

#posts 0

#followers 0

#follows 0

fake 0

dtype: int64

df.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 576 entries, 0 to 575

Data columns (total 12 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 profile pic 576 non-null int64

1 nums/length username 576 non-null float64

2 fullname words 576 non-null int64

3 nums/length fullname 576 non-null float64

4 name==username 576 non-null int64

5 description length 576 non-null int64

6 external URL 576 non-null int64

7 private 576 non-null int64

8 #posts 576 non-null int64

9 #followers 576 non-null int64

10 #follows 576 non-null int64

11 fake 576 non-null int64

dtypes: float64(2), int64(10)

memory usage: 54.1 KB

df['profile pic'].unique()

array([1, 0], dtype=int64)

df['nums/length username'].unique()

array([0.27, 0. , 0.1 , 0.29, 0.13, 0.18, 0.33, 0.12, 0.19, 0.2 , 0.06,

0.44, 0.24, 0.14, 0.36, 0.22, 0.08, 0.15, 0.09, 0.45, 0.38, 0.11,

0.17, 0.31, 0.3 , 0.07, 0.43, 0.5 , 0.25, 0.57, 0.88, 0.55, 0.8 ,

0.4 , 0.41, 0.67, 0.89, 0.16, 0.58, 0.64, 0.42, 0.21, 0.71, 0.83,

0.47, 0.91, 0.28, 0.54, 0.46, 0.73, 0.6 , 0.86, 0.62, 0.92])

df['fullname words'].unique()

array([ 0, 2, 1, 4, 3, 6, 12, 5, 10], dtype=int64)

df['nums/length fullname'].unique()

array([0. , 0.12, 0.1 , 0.08, 0.24, 0.25, 0.33, 0.14, 0.22, 0.43, 0.4 ,

0.36, 0.29, 0.11, 0.31, 0.89, 0.2 , 0.38, 0.27, 0.44, 1. , 0.5 ,

0.18, 0.57, 0.46])

df['name==username'].unique()

array([0, 1], dtype=int64)

df['description length'].unique()

array([ 53, 44, 0, 82, 81, 50, 71, 40, 54, 103, 98, 46, 48,

63, 106, 35, 30, 27, 109, 132, 126, 122, 138, 56, 9, 134,

2, 23, 93, 4, 1, 91, 57, 108, 12, 3, 39, 68, 129,

64, 42, 70, 74, 8, 28, 18, 36, 11, 29, 24, 21, 34,

59, 15, 16, 73, 26, 55, 140, 113, 38, 89, 123, 33, 5,

150, 149, 148, 19, 133, 43, 37, 87, 95, 117, 58, 62, 137,

14, 131, 10, 72, 51, 13, 105, 67, 20, 86, 96, 17, 61,

112, 101, 32, 80, 146, 6, 49, 120, 25, 139, 31, 7, 22],

dtype=int64)

df['external URL'].unique()

array([0, 1], dtype=int64)

df['private'].unique()

array([0, 1], dtype=int64)

df['#posts'].unique()

array([ 32, 286, 13, 679, 6, 344, 16, 33, 72, 213, 648,

76, 298, 117, 487, 254, 59, 1570, 378, 526, 228, 35,

281, 285, 148, 57, 17, 511, 230, 15, 980, 53, 111,

719, 1164, 497, 18, 50, 74, 8, 7389, 420, 433, 156,

4494, 751, 4, 27, 91, 262, 274, 271, 713, 200, 12,

26, 75, 94, 63, 69, 19, 100, 661, 149, 22, 400,

122, 77, 5, 3, 106, 14, 172, 38, 227, 221, 580,

40, 101, 157, 197, 61, 698, 49, 85, 58, 232, 20,

98, 559, 189, 388, 28, 775, 205, 209, 334, 9, 416,

1, 711, 114, 107, 7, 21, 65, 10, 137, 571, 24,

328, 161, 280, 92, 31, 0, 25, 921, 1020, 301, 158,

43, 60, 220, 1159, 396, 2, 131, 36, 11, 252, 83,

126, 663, 64, 664, 130, 917, 142, 165, 80, 81, 373,

56, 93, 192, 145, 135, 222, 119, 201, 112, 54, 133,

70, 403, 990, 411, 217, 389, 144, 78, 86, 240, 44,

62, 52, 247, 204, 51, 108, 353, 560, 95, 89, 34,

283, 327, 42, 103, 241, 1232, 30, 141, 39, 124, 150,

29, 84, 299, 37, 23, 102], dtype=int64)

df['#followers'].unique()

array([ 1000, 2740, 159, 414, 151, 669987,

122, 1078, 1824, 12945, 9884, 1188,

945, 12033, 1962, 50374, 7007, 1128,

34670, 2338, 3516, 1809, 427, 759,

15338538, 109, 536, 121354, 2284, 186,

687, 966, 177, 744, 542073, 5315651,

267, 691, 120, 105, 890969, 361853,

3678, 92192, 12397719, 380510, 132, 162,

369, 1476, 1798, 2118, 812, 7217,

313, 64, 1759, 404, 1843, 320377,

108, 384, 60, 802, 51145, 1582,

223, 18842, 10240, 539, 399, 581,

166, 417, 266, 33, 494, 178,

470, 807, 17303, 1439, 91446, 824,

741, 1267, 4594, 1135, 1926, 1068,

815, 565, 2556, 1312, 699, 4328,

2487, 673, 730, 59, 289, 19,

3551, 19512, 2756, 5406, 459, 218,

796, 1113, 138, 205, 331, 748,

490, 456, 971, 497, 99, 193,

492, 167, 916, 196, 765, 45,

634, 1383, 650, 484, 200, 192,

553, 27477, 464, 1057, 413, 389,

505, 941, 2598, 622, 2719, 216,

881, 870, 265, 1204, 655, 1662,

14222, 483, 408, 303, 125, 229,

357, 137, 255, 87, 326946, 114,

1247, 585, 135, 722, 714, 39867,

533, 1158, 45834, 876, 3003, 1298,

3800, 1358, 6741307, 791, 732075, 373,

309, 244, 67, 984, 751, 781,

1761, 318, 5282, 228, 393, 875,

173, 3896490, 106, 206, 259, 1013,

738, 1008, 2441, 416, 516, 8711,

433, 18515, 70, 5863, 1677, 617,

31182, 1152, 8578, 4347, 319, 189,

743, 11204, 419, 81, 947, 541,

392, 4177, 272, 425, 150, 711,

89, 742, 96, 77, 195, 9,

10794, 104, 355, 300, 139, 13,

606, 428, 1261, 68, 205558, 668,

1456, 410, 298, 254, 1070, 1167,

335, 346, 1746, 268, 537, 805,

1504, 380, 257, 1775, 1051, 220,

728, 406, 37, 56, 90, 271,

1, 158, 39, 0, 12, 10,

31, 5, 18, 6, 47, 4,

107, 8, 48, 2, 51, 26,

76, 165, 115, 24, 40, 32,

21, 79, 49, 15, 316, 3,

221, 181, 25, 7, 133, 864,

73, 184, 161, 42, 279, 219,

34, 38, 23, 83, 82, 140,

124, 11, 446, 589, 27, 143,

16, 57, 1031, 46, 834, 43,

17, 182, 53, 35, 119, 818,

102, 576, 66, 310, 88, 61,

358, 20, 50, 78, 30, 85,

92, 75, 22, 55, 86, 14,

52, 136, 207, 356, 44, 3033,

210, 1489, 201, 351, 156, 97,

2346, 332, 65, 72, 58, 126],

dtype=int64)

df['#follows'].unique()

array([ 955, 533, 98, 651, 126, 150, 177, 76, 2713, 813, 1173,

365, 583, 248, 2701, 900, 289, 694, 1878, 776, 999, 416,

470, 956, 61, 179, 665, 176, 130, 174, 1517, 952, 170,

967, 674, 2703, 328, 680, 112, 11, 1359, 16, 8, 0,

183, 208, 546, 666, 461, 1109, 432, 761, 376, 261, 643,

283, 598, 228, 97, 447, 100, 151, 528, 1882, 266, 744,

1255, 639, 452, 568, 163, 362, 324, 37, 998, 245, 288,

675, 256, 395, 360, 629, 526, 489, 1440, 899, 1713, 1410,

1925, 748, 469, 1074, 935, 599, 418, 438, 413, 55, 222,

20, 173, 591, 638, 589, 390, 75, 1155, 1854, 164, 333,

4659, 1093, 2047, 132, 689, 178, 1142, 209, 424, 80, 719,

7500, 703, 3296, 270, 65, 610, 7202, 1039, 524, 138, 806,

503, 1208, 802, 111, 475, 1061, 305, 375, 72, 371, 633,

1016, 1065, 7399, 1216, 2928, 635, 417, 101, 383, 96, 535,

399, 3, 446, 387, 1196, 1364, 232, 1159, 4664, 1060, 3932,

280, 529, 455, 407, 278, 127, 456, 363, 311, 364, 172,

149, 1223, 905, 523, 652, 238, 502, 754, 373, 351, 133,

32, 1371, 996, 544, 517, 396, 292, 463, 345, 584, 1000,

67, 157, 716, 272, 414, 335, 331, 124, 271, 268, 582,

701, 237, 3678, 321, 1486, 917, 673, 121, 653, 50, 107,

13, 3164, 15, 569, 137, 488, 372, 77, 430, 545, 2187,

87, 53, 609, 555, 1072, 618, 300, 426, 1631, 370, 1002,

52, 119, 462, 346, 1200, 264, 34, 323, 81, 1213, 294,

408, 22, 114, 31, 445, 309, 64, 853, 23, 5, 18,

213, 10, 58, 4, 279, 44, 17, 42, 60, 6, 215,

48, 46, 601, 168, 230, 49, 236, 35, 767, 79, 91,

41, 229, 1165, 1, 30, 244, 167, 86, 129, 47, 750,

39, 3646, 1124, 155, 106, 171, 108, 26, 68, 139, 28,

38, 69, 319, 29, 420, 733, 45, 249, 56, 25, 71,

104, 99, 40, 1762, 82, 2980, 63, 500, 135, 192, 36,

2, 181, 146, 92, 4118, 392, 221, 227, 304, 123, 66,

303, 90, 336, 88, 136, 24, 295, 74, 233, 109, 474,

161, 894, 59, 2330, 433, 1269, 14, 595, 9, 1990, 7,

792, 747, 83, 403, 27, 1029, 1417, 2426, 828, 12, 2176,

240, 153, 825, 1528, 1543, 3715, 812, 2663, 423, 477, 51,

7272, 95, 19, 1333, 542, 162, 434, 347, 242, 860, 698,

596, 339, 73, 487], dtype=int64)

df['fake'].unique()

array([0, 1], dtype=int64)

import seaborn as sns

plt.style.use('seaborn')

df['fake'].hist(bins=20)

C:\Users\IT Support\AppData\Local\Temp\ipykernel\_13044\1213997271.py:1: MatplotlibDeprecationWarning: The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no longer correspond to the styles shipped by seaborn. However, they will remain available as 'seaborn-v0\_8-<style>'. Alternatively, directly use the seaborn API instead.

plt.style.use('seaborn')

<Axes: >

plt.style.use('seaborn')

df['private'].hist(bins=20)

C:\Users\IT Support\AppData\Local\Temp\ipykernel\_13044\1295440802.py:1: MatplotlibDeprecationWarning: The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no longer correspond to the styles shipped by seaborn. However, they will remain available as 'seaborn-v0\_8-<style>'. Alternatively, directly use the seaborn API instead.

plt.style.use('seaborn')

<Axes: >

plt.style.use('seaborn')

df['private'].hist(bins=20)

C:\Users\IT Support\AppData\Local\Temp\ipykernel\_13044\845631239.py:1: MatplotlibDeprecationWarning: The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no longer correspond to the styles shipped by seaborn. However, they will remain available as 'seaborn-v0\_8-<style>'. Alternatively, directly use the seaborn API instead.

plt.style.use('seaborn')

<Axes: >

plt.style.use('seaborn')

df['profile pic'].hist(bins=20)

C:\Users\IT Support\AppData\Local\Temp\ipykernel\_13044\3758939069.py:1: MatplotlibDeprecationWarning: The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no longer correspond to the styles shipped by seaborn. However, they will remain available as 'seaborn-v0\_8-<style>'. Alternatively, directly use the seaborn API instead.

plt.style.use('seaborn')

<Axes: >

df.dtypes

profile pic int64

nums/length username float64

fullname words int64

nums/length fullname float64

name==username int64

description length int64

external URL int64

private int64

#posts int64

#followers int64

#follows int64

fake int64

dtype: object

plt.rcParams["figure.figsize"] = (30,10)

import seaborn as sns

sns.distplot(df["#follows"])

df['#follows'].skew()

C:\Users\IT Support\AppData\Local\Temp\ipykernel\_13044\2582022536.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with

similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see

https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

sns.distplot(df["#follows"])

4.724967697205058

sns.distplot(df["#followers"])

df['#followers'].skew()

C:\Users\IT Support\AppData\Local\Temp\ipykernel\_13044\905562628.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with

similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see

https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

sns.distplot(df["#followers"])

13.67904486141773

sns.distplot(df["#posts"])

df['#posts'].skew()

C:\Users\IT Support\AppData\Local\Temp\ipykernel\_13044\754962500.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with

similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see

https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

sns.distplot(df["#posts"])

12.986195887205703

sns.distplot(df["private"])

df['private'].skew()

C:\Users\IT Support\AppData\Local\Temp\ipykernel\_13044\867369801.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with

similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see

https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

sns.distplot(df["private"])

0.4872320413461685

sns.distplot(df["external URL"])

df['external URL'].skew()

C:\Users\IT Support\AppData\Local\Temp\ipykernel\_13044\3826985736.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with

similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see

https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

sns.distplot(df["external URL"])

2.3997126920306884

sns.distplot(df["description length"])

df['description length'].skew()

C:\Users\IT Support\AppData\Local\Temp\ipykernel\_13044\1028226637.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with

similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see

https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

sns.distplot(df["description length"])

1.8668004695721174

sns.distplot(df["nums/length fullname"])

df['nums/length fullname'].skew()

C:\Users\IT Support\AppData\Local\Temp\ipykernel\_13044\1596414153.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with

similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see

https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

sns.distplot(df["nums/length fullname"])

4.436645238006572

sns.distplot(df["fullname words"])

df['fullname words'].skew()

C:\Users\IT Support\AppData\Local\Temp\ipykernel\_13044\2893400398.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with

similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see

https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

sns.distplot(df["fullname words"])

3.324440164721317

sns.distplot(df["nums/length username"])

df['nums/length username'].skew()

C:\Users\IT Support\AppData\Local\Temp\ipykernel\_13044\3195484667.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with

similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see

https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

sns.distplot(df["nums/length username"])

1.2629019102928685

df.dtypes

profile pic int64

nums/length username float64

fullname words int64

nums/length fullname float64

name==username int64

description length int64

external URL int64

private int64

#posts int64

#followers int64

#follows int64

fake int64

dtype: object

df.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 576 entries, 0 to 575

Data columns (total 12 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 profile pic 576 non-null int64

1 nums/length username 576 non-null float64

2 fullname words 576 non-null int64

3 nums/length fullname 576 non-null float64

4 name==username 576 non-null int64

5 description length 576 non-null int64

6 external URL 576 non-null int64

7 private 576 non-null int64

8 #posts 576 non-null int64

9 #followers 576 non-null int64

10 #follows 576 non-null int64

11 fake 576 non-null int64

dtypes: float64(2), int64(10)

memory usage: 54.1 KB

# Input = ['Make','Vehicle class','Engine Size(L)','Cylinders','Transmission','Fuel Type', 'Fuel consumption city', 'Fuel consumption hwy','Fuel consumption comb','Fuel consumption comb mpg']

X = df.drop(['fake'], axis=1).values

y = df['fake'].values

X.shape

(576, 11)

y

array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1], dtype=int64)

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.2,random\_state=10)

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import RobustScaler

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification\_report, confusion\_matrix

rfc = RandomForestClassifier(n\_estimators=80, random\_state=23)

rfc.fit(x\_train,y\_train)

RandomForestClassifier

RandomForestClassifier(n\_estimators=80, random\_state=23)

rfc.score(x\_train,y\_train)

1.0

from sklearn.metrics import accuracy\_score

y\_pred = rfc.predict(x\_test )

accuracy\_score(y\_pred,y\_test)

0.9396551724137931

import sklearn.metrics

print(sklearn.metrics.classification\_report(y\_test, y\_pred))

precision recall f1-score support

0 0.90 0.98 0.94 58

1 0.98 0.90 0.94 58

accuracy 0.94 116

macro avg 0.94 0.94 0.94 116

weighted avg 0.94 0.94 0.94 116

y\_pred = rfc.predict(x\_test )

y\_true=y\_test

from sklearn.metrics import confusion\_matrix

cm=confusion\_matrix(y\_true,y\_pred)

cm

array([[57, 1],

[ 6, 52]], dtype=int64)

import seaborn as sns

import matplotlib.pyplot as plt

f, ax=plt.subplots(figsize=(5,5))

sns.heatmap(cm,annot=True,linewidths=0.5,linecolor="red",fmt=".0f",ax=ax)

plt.xlabel("y\_pred")

plt.ylabel("y\_true")

plt.show()

import pickle

pickle.dump(rfc,open('random\_fake.pkl','wb'))

random = pickle.load(open('random\_fake.pkl','rb'))

from sklearn.tree import DecisionTreeClassifier

clf\_gini = DecisionTreeClassifier(criterion='gini', max\_depth=3, random\_state=0)

clf\_gini.fit(x\_train,y\_train)

DecisionTreeClassifier

DecisionTreeClassifier(max\_depth=3, random\_state=0)

clf\_gini.score(x\_train,y\_train)

0.9217391304347826

from sklearn.metrics import accuracy\_score

y\_pred = clf\_gini.predict(x\_test )

accuracy\_score(y\_pred,y\_test)

0.9224137931034483

print(sklearn.metrics.classification\_report(y\_test, y\_pred))

precision recall f1-score support

0 0.89 0.97 0.93 58

1 0.96 0.88 0.92 58

accuracy 0.92 116

macro avg 0.93 0.92 0.92 116

weighted avg 0.93 0.92 0.92 116

y\_pred = clf\_gini.predict(x\_test )

y\_true=y\_test

from sklearn.metrics import confusion\_matrix

cm=confusion\_matrix(y\_true,y\_pred)

cm

array([[56, 2],

[ 7, 51]], dtype=int64)

import seaborn as sns

import matplotlib.pyplot as plt

f, ax=plt.subplots(figsize=(5,5))

sns.heatmap(cm,annot=True,linewidths=0.5,linecolor="red",fmt=".0f",ax=ax)

plt.xlabel("y\_pred")

plt.ylabel("y\_true")

plt.show()

import pickle

pickle.dump(clf\_gini,open('decision\_fake.pkl','wb'))

clf\_gini = pickle.load(open('decision\_fake.pkl','rb'))