Assignment 4.1

September 30, 2024

[1]: import pandas as pd

```
import numpy as np
from IPython.display import display, HTML
# Load the data
df = pd.read csv(
    "/Users/gabrielmancillas/Desktop/ADS 505-01/Mod 04/Assignment 4.1/
 →Fundraising.csv"
dff = pd.read_csv(
    "/Users/gabrielmancillas/Desktop/ADS 505-01/Mod 04/Assignment 4.1/
 ⇔FutureFundraising.csv"
# Display both DataFrames side by side
display(HTML("<h3>Fundraising DataFrame</h3>"))
display(df.head())
display(HTML("<h3>Future Fundraising DataFrame</h3>"))
display(dff.head())
/var/folders/jw/4t4swxld5c5f_5xhv0_bzbr00000gn/T/ipykernel_97635/3847261029.py:1
: DeprecationWarning:
Pyarrow will become a required dependency of pandas in the next major release of
pandas (pandas 3.0),
(to allow more performant data types, such as the Arrow string type, and better
interoperability with other libraries)
but was not found to be installed on your system.
If this would cause problems for you,
please provide us feedback at https://github.com/pandas-dev/pandas/issues/54466
  import pandas as pd
<IPython.core.display.HTML object>
  Row Id Row Id. zipconvert_2 zipconvert_3 zipconvert_4 zipconvert_5 \
       1
0
                17
                               0
                                             1
                                                           0
        2
                25
                                             0
                                                           0
                                                                         0
1
                               1
2
        3
                29
                               0
                                             0
                                                           0
                                                                          1
```

3 4	4 5	38 40	0		0 1	0 0	1 0	
0	homeowner	dummy NU	JMCHLD INCO 1 1	ME gender o 5	1	IC15 NUMP	ROM \ 74 46	
2		0	2	5	0 1	13	32	
3		1	1	3	0	4	94	
4		1	1	4	0	7	20	
0	RAMNTALL 102.0	MAXRAMNT 6.0		totalmonths	TIMELAG	AVGGIFT		\
1	94.0	12.0	12.0	34	6		1	
2	30.0	10.0	5.0	29	7		1	
3	177.0	10.0	8.0	30	3		0	
4	23.0	11.0	11.0	30	6		0	
4	23.0	11.0	11.0	30	0	1.000001	U	
0	TARGET_D 5.0							
1	10.0							
2	5.0							
3	0.0							
4	0.0							
	rows x 24							
<i< td=""><td>Python.cor</td><td>e.display</td><td>.HTML object</td><td>></td><td></td><td></td><td></td><td></td></i<>	Python.cor	e.display	.HTML object	>				
	Row Id R	ow Id. zi	ipconvert_2	zipconvert_	3 zipco	nvert_4 zi	pconvert_5	
0	1	3	•				bconver c_o	\
1	2	U	0		1	0	0	\
2	2	4	0		0	0 1	="	\
	3						0	\
3		4	0		0	1	0	\
	3	4 5	0		0 0	1 0	0 0 1	\
3	3 4	4 5 1 4 dummy NU	0 0 0 0 0	•	0 0 0 0 0	1 0 0 1 IC15 NUMP	0 0 1 0 0	\
3 4	3 4 5	4 5 1 4 dummy NU 1	0 0 0 0 0 0 JMCHLD INCO	1	0 0 0 0 0 ummy 1	1 0 0 1 1 IC15 NUMP	0 0 1 0 0 0	\
3 4 0 1	3 4 5	4 5 1 4 dummy NU 1 0 0	O O O O JMCHLD INCO	1 2	0 0 0 0 0 dummy 1 1	1 0 0 1 1 IC15 NUMP	0 0 1 0 0 0 ROM \ 42 21	\
3 4 0 1 2	3 4 5	4 5 1 4 dummy NU 1 0 0	O O O O JMCHLD INCO: 1 1 1	1 2 1	0 0 0 0 0 dummy 1 0	1 0 0 1 IC15 NUMP: 3 4 10	0 0 1 0 0 0 ROM \ 42 21 61	\
3 4 0 1 2 3	3 4 5	4 5 1 4 4 dummy NU 1 0 0 0 1 1	0 0 0 0 0 JMCHLD INCO	1 2 1 4	0 0 0 0 0 dummy 1 1 0 0	1 0 0 1 IC15 NUMP 3 4 10 21	0 0 1 0 0 0 ROM \ 42 21 61 32	\
3 4 0 1 2	3 4 5	4 5 1 4 dummy NU 1 0 0	O O O O JMCHLD INCO: 1 1 1	1 2 1	0 0 0 0 0 dummy 1 0	1 0 0 1 IC15 NUMP: 3 4 10	0 0 1 0 0 0 ROM \ 42 21 61	\
3 4 0 1 2 3 4	3 4 5 homeowner	4 5 1 4 4 dummy NU 1 0 0 1 1 1 MAXRAMNT	0 0 0 0 JMCHLD INCO 1 1 1 1 1 1 LASTGIFT	1 2 1 4 7 totalmonths	0 0 0 0 dummy 1 0 0 1	1 0 0 1 IC15 NUMP 3 4 10 21 1	0 0 1 0 0 0 ROM \ 42 21 61 32 47	\
3 4 0 1 2 3 4	3 4 5 homeowner	4 5 1 4 4 dummy NU 1 0 0 0 1 1 1 MAXRAMNT 29.0	0 0 0 0 0 JMCHLD INCO 1 1 1 1 1 1 LASTGIFT 15.0	1 2 1 4 7 totalmonths 17	0 0 0 0 dummy 1 0 1 TIMELAG 8	1 0 0 1 IC15 NUMP 3 4 10 21 1 AVGGIFT 15.333333	0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	\
3 4 0 1 2 3 4	3 4 5 homeowner RAMNTALL 92.0 30.0	4 5 1 4 4 dummy NU 1 0 0 0 1 1 1 MAXRAMNT 29.0 20.0	0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 2 1 20.0	1 2 1 4 7 totalmonths 17 33	0 0 0 0 0 dummy 1 0 1 TIMELAG 8	1 0 0 1 IC15 NUMP 3 4 10 21 1 AVGGIFT 15.333333 15.0000000	0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	\
3 4 0 1 2 3 4	3 4 5 homeowner RAMNTALL 92.0 30.0 220.0	4 5 1 4 4 dummy NU 1 0 0 0 1 1 1 MAXRAMNT 29.0 20.0 35.0	0 0 0 0 0 JMCHLD INCO 1 1 1 1 1 1 1 1 20.0 20.0 25.0	1 2 1 4 7 totalmonths 17 33 31	0 0 0 0 0 dummy 1 0 1 TIMELAG 8 9	1 0 0 1 IC15 NUMP 3 4 10 21 1 AVGGIFT 15.333333 15.000000 24.444444	0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	\
3 4 0 1 2 3 4	3 4 5 homeowner RAMNTALL 92.0 30.0	4 5 1 4 4 dummy NU 1 0 0 0 1 1 1 MAXRAMNT 29.0 20.0	0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 2 1 20.0	1 2 1 4 7 totalmonths 17 33	0 0 0 0 0 dummy 1 0 1 TIMELAG 8	1 0 0 1 IC15 NUMP 3 4 10 21 1 1 AVGGIFT 15.333333 15.000000 24.444444 13.666667	0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	\

```
TARGET D
    0
            NaN
            NaN
    1
    2
            NaN
            NaN
    3
    4
            NaN
    [5 rows x 24 columns]
[2]: import pandas as pd
     import numpy as np
     from IPython.display import display, HTML
     # Load the data
     df = pd.read_csv(
         "/Users/gabrielmancillas/Desktop/ADS 505-01/Mod 04/Assignment 4.1/

→Fundraising.csv"

     dff = pd.read_csv(
         "/Users/gabrielmancillas/Desktop/ADS 505-01/Mod 04/Assignment 4.1/

→FutureFundraising.csv"

     )
     # 1. Understand the Data
     display(HTML("<h3>First 5 rows of the Fundraising dataset:</h3>"))
     display(df.head())
     display(HTML("<h3>Data types and missing values in Fundraising dataset:</h3>"))
     display(df.info())
     display(HTML("<h3>Summary statistics of Fundraising dataset:</h3>"))
     display(df.describe())
     # 2. Handle Missing Values
     display(HTML("<h3>Missing values per column in Fundraising dataset:</h3>"))
     display(df.isnull().sum())
     # Fill missing values with the mean
     df.fillna(df.mean(), inplace=True)
     # Handle the dff dataframe
     display(HTML("<h3>First 5 rows of the Future Fundraising dataset:</h3>"))
     display(dff.head())
     display(HTML("<h3>Data types and missing values in Future Fundraising dataset:/
     display(dff.info())
```

```
display(HTML("<h3>Summary statistics of Future Fundraising dataset:</h3>"))
display(dff.describe())
display(HTML("<h3>Missing values per column in Future Fundraising dataset:/
  ⇔h3>"))
display(dff.isnull().sum())
# Fill missing values in dff with the mean
dff.fillna(dff.mean(), inplace=True)
<IPython.core.display.HTML object>
   Row Id Row Id. zipconvert_2 zipconvert_3 zipconvert_4 zipconvert_5 \
0
        1
                17
                                0
                                              1
        2
                25
1
                                1
                                              0
                                                             0
                                                                           0
2
        3
                29
                                0
                                              0
                                                             0
                                                                           1
3
        4
                                              0
                                                                           1
                38
                                0
                                                             0
4
        5
                40
                                0
                                              1
                                                                           0
  homeowner dummy
                    NUMCHLD INCOME
                                      gender dummy
                                                       IC15
                                                            NUMPROM
0
                                   5
                                                                   74
                 1
                           1
                                                 1
                                                           1
                 1
                           1
                                   1
                                                           4
                                                                   46
1
                                                 0
                 0
                           2
2
                                   5
                                                 1
                                                          13
                                                                   32
3
                 1
                           1
                                   3
                                                 0
                                                           4
                                                                   94
4
                           1
                                   4
                                                           7
                 1
                                                                   20
  RAMNTALL MAXRAMNT LASTGIFT totalmonths TIMELAG
                                                          AVGGIFT TARGET B
0
      102.0
                  6.0
                             5.0
                                                     3 4.857143
                                                                          1
1
       94.0
                 12.0
                            12.0
                                           34
                                                     6 9.400000
                                                                          1
2
       30.0
                 10.0
                            5.0
                                           29
                                                     7 4.285714
                                                                          1
3
      177.0
                 10.0
                            8.0
                                           30
                                                     3 7.080000
                                                                          0
4
       23.0
                 11.0
                           11.0
                                                     6 7.666667
                                                                          0
                                           30
   TARGET D
        5.0
0
1
       10.0
2
        5.0
3
        0.0
4
        0.0
[5 rows x 24 columns]
<IPython.core.display.HTML object>
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3120 entries, 0 to 3119
Data columns (total 24 columns):
     Column
                      Non-Null Count Dtype
```

0	Row Id	3120 non-null	int64
1	Row Id.	3120 non-null	int64
2	zipconvert_2	3120 non-null	int64
3	zipconvert_3	3120 non-null	int64
4	zipconvert_4	3120 non-null	int64
5	zipconvert_5	3120 non-null	int64
6	homeowner dummy	3120 non-null	int64
7	NUMCHLD	3120 non-null	int64
8	INCOME	3120 non-null	int64
9	gender dummy	3120 non-null	int64
10	WEALTH	3120 non-null	int64
11	HV	3120 non-null	int64
12	Icmed	3120 non-null	int64
13	Icavg	3120 non-null	int64
14	IC15	3120 non-null	int64
15	NUMPROM	3120 non-null	int64
16	RAMNTALL	3120 non-null	float64
17	MAXRAMNT	3120 non-null	float64
18	LASTGIFT	3120 non-null	float64
19	totalmonths	3120 non-null	int64
20	TIMELAG	3120 non-null	int64
21	AVGGIFT	3120 non-null	float64
22	TARGET_B	3120 non-null	int64
23	TARGET_D	3120 non-null	float64

dtypes: float64(5), int64(19) memory usage: 585.1 KB

None

<IPython.core.display.HTML object>

	Row Id	Row Id. z	zipconvert_2	zipconvert_3	zipconvert_4 \	\
count	3120.000000	3120.000000	3120.000000	3120.000000	3120.000000	
mean	1560.500000	11615.770833	0.214423	0.185256	0.214423	
std	900.810746	6698.678131	0.410487	0.388568	0.410487	
min	1.000000	17.000000	0.000000	0.00000	0.000000	
25%	780.750000	5820.750000	0.000000	0.00000	0.000000	
50%	1560.500000	11735.500000	0.000000	0.00000	0.000000	
75%	2340.250000	17435.750000	0.000000	0.00000	0.000000	
max	3120.000000	23293.000000	1.000000	1.000000	1.000000	
	zipconvert_5	homeowner dumm	ny NUMCHL	D INCOME	gender dummy	\
count	3120.000000	3120.00000	00 3120.00000	0 3120.000000	3120.000000	
mean	0.384615	0.77019	1.06923	1 3.893910	0.609295	
std	0.486582	0.42077	77 0.34768	8 1.636186	0.487987	
min	0.000000	0.00000	1.00000	0 1.000000	0.000000	
25%	0.000000	1.00000	1.00000	0 3.000000	0.000000	
50%	0.000000	1.00000	1.00000	0 4.000000	1.000000	

75% max		1.000000		0000 0000	1.0000		00000		
		IC	15 NUMPR	OM :	RAMNTALL	MAXRA	MNT	LASTGIFT	\
count		3120.0000	00 3120.0000	00 312	0.000000	3120.000	000	3120.000000	
mean		14.7028	85 49.0894	23 11	0.399875	16.651	.397	13.522917	
std		12.0798	82 22.7171	30 14	7.299933	22.223	3521	10.581439	
min		0.0000	00 11.0000	00 1	5.000000	5.000	000	0.000000	
25%		5.0000	00 29.0000	00 4	5.000000	10.000	000	7.000000	
50%		12.0000	00 48.0000	00 8	1.000000	15.000	000	10.000000	
75%		21.0000	00 65.0000	00 13	4.625000	20.000	000	16.000000	
max		90.0000	00 157.0000	00 567	4.900000	1000.000	000	219.000000	
	to	talmonths	TIMELAG	AV	GGIFT	TARGET_B	T	ARGET_D	
count	31	20.000000	3120.000000	3120.0	00000 3	120.00000	3120	0.00000	
mean		31.136859	6.861859	10.6	90713	0.50000	6	3.499612	
std		4.132952	5.561209	7.4	43980	0.50008	10	.597849	
min		17.000000	0.000000	2.1	38889	0.00000	0	0.00000	
25%		29.000000	3.000000	6.3	56092	0.00000	0	0.00000	
50%		31.000000	5.000000	9.0	00000	0.50000	0	.500000	
75%		34.000000	9.000000	12.8	11652	1.00000	10	0.00000	
max		37.000000	77.000000	122.1	66667	1.00000	200	0.00000	

[8 rows x 24 columns]

<IPython.core.display.HTML object>

Row Id 0 Row Id. 0 zipconvert_2 0 zipconvert_3 0 zipconvert_4 0 zipconvert_5 0 homeowner dummy NUMCHLD 0 INCOME 0 gender dummy 0 WEALTH 0 HV0 Icmed 0 Icavg 0 IC15 0 NUMPROM 0 RAMNTALL 0 0 MAXRAMNT LASTGIFT 0 totalmonths 0 TIMELAG 0 0 AVGGIFT

TARGET_B 0
TARGET_D 0

dtype: int64

<IPython.core.display.HTML object>

	Row Id	Row Id.	zipconvert_2	zipconvert_3	zipconvert_4	zipconvert_5	\
0	1	3	0	1	0	0	
1	2	4	0	0	1	0	
2	3	5	0	0	0	1	
3	4	1	0	0	0	0	
4	5	4	0	0	1	0	

	homeowner	dummy	NUMCHLD	INCOME	gender dummy	•••	IC15	NUMPROM	\
0		1	1	1	1		3	42	
1		0	1	2	1		4	21	
2		0	1	1	0		10	61	
3		1	1	4	0		21	32	
4		1	1	7	1		1	47	

	RAMNTALL	MAXRAMNT	LASTGIFT	totalmonths	TIMELAG	AVGGIFT	TARGET_B	\
0	92.0	29.0	15.0	17	8	15.333333	NaN	
1	30.0	20.0	20.0	33	9	15.000000	NaN	
2	220.0	35.0	25.0	31	9	24.44444	NaN	
3	41.0	19.0	19.0	31	13	13.666667	NaN	
4	46.0	10.0	10.0	28	8	5.750000	NaN	

TARGET_D
0 NaN
1 NaN
2 NaN
3 NaN
4 NaN

[5 rows x 24 columns]

<IPython.core.display.HTML object>

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 24 columns):

#	Column	Non-Null Count	Dtype
0	Row Id	2000 non-null	int64
1	Row Id.	2000 non-null	int64
2	zipconvert_2	2000 non-null	int64
3	zipconvert_3	2000 non-null	int64
4	zipconvert_4	2000 non-null	int64
5	zipconvert_5	2000 non-null	int64
6	homeowner dummy	2000 non-null	int64

7	NUMCHLD	2000 non-null	int64
8	INCOME	2000 non-null	int64
9	gender dummy	2000 non-null	int64
10	WEALTH	2000 non-null	int64
11	HV	2000 non-null	int64
12	Icmed	2000 non-null	int64
13	Icavg	2000 non-null	int64
14	IC15	2000 non-null	int64
15	NUMPROM	2000 non-null	int64
16	RAMNTALL	2000 non-null	float64
17	MAXRAMNT	2000 non-null	float64
18	LASTGIFT	2000 non-null	float64
19	totalmonths	2000 non-null	int64
20	TIMELAG	2000 non-null	int64
21	AVGGIFT	2000 non-null	float64
22	TARGET_B	0 non-null	float64
23	TARGET_D	0 non-null	float64
J	£1+C1(C)	+C1(10)	

dtypes: float64(6), int64(18)

memory usage: 375.1 KB

None

<IPython.core.display.HTML object>

	Row Id	Row Id.	zipconvert_2	zipconvert_3	<pre>zipconvert_4 \</pre>	
count	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000	
mean	1000.500000	3.482500	0.237500	0.168500	0.230000	
std	577.494589	1.300592	0.425658	0.374403	0.420938	
min	1.000000	1.000000	0.000000	0.000000	0.000000	
25%	500.750000	2.000000	0.000000	0.000000	0.000000	
50%	1000.500000	4.000000	0.000000	0.000000	0.000000	
75%	1500.250000	5.000000	0.000000	0.000000	0.000000	
max	2000.000000	5.000000	1.000000	1.000000	1.000000	
	zipconvert_5	homeowner d	ummy NUMC	HLD INCOM	ME gender dummy	\
count	2000.00000	2000.0	0000 2000.000	2000.00000	2000.000000	
mean	0.30450	0.7	7850 1.052	3.82150	0.595500	
std	0.46031	0.4	1536 0.278	091 1.63859	0.490918	
min	0.00000	0.0	0000 1.000	1.00000	0.00000	
25%	0.00000	1.0	0000 1.000	2.75000	0.00000	
50%	0.00000	1.0	0000 1.000	000 4.00000	1.00000	
75%	1.00000	1.0	0000 1.000	5.00000	1.00000	
max	1.00000	1.0	0000 5.000	7.00000	1.00000	
	IC	15 NUMPR	OM RAMNTAL	L MAXRAMNT	LASTGIFT \	
count	2000.0000	00 2000.0000	00 2000.00000	0 2000.000000	2000.000000	
mean	15.1770	00 47.3630	00 103.00542	5 19.270305	16.478900	
std	12.5779	88 22.9639	91 98.84494	3 11.752492	9.714386	
min	0.0000	00 4.0000	15.00000	5.000000	0.000000	

```
25%
             5.750000
                          27.000000
                                        42.000000
                                                       14.000000
                                                                     10.000000
50%
            12.000000
                          47.000000
                                        80.000000
                                                       17.000000
                                                                     15.000000
75%
            22.000000
                          64.000000
                                       132.000000
                                                       22.000000
                                                                     20.000000
            87.000000
                         148.000000
                                      1526.000000
                                                     200.000000
                                                                   105.000000
max
                                                 TARGET_B
                                                            TARGET D
       totalmonths
                         TIMELAG
                                       AVGGIFT
                                                      0.0
count
        2000.00000
                     2000.000000
                                   2000.000000
                                                                 0.0
                                                      NaN
mean
          31.29300
                        7.678000
                                     13.051707
                                                                 NaN
std
           4.04244
                        5.652459
                                      7.982889
                                                      NaN
                                                                 NaN
          17.00000
                                                      NaN
                                                                 NaN
min
                        0.000000
                                      1.636364
25%
          29.00000
                                                      NaN
                                                                 {\tt NaN}
                        4.000000
                                      8.408824
50%
          31.00000
                        6.000000
                                                      NaN
                                                                 NaN
                                     11.600000
75%
          33.00000
                       10.000000
                                     15.211806
                                                      {\tt NaN}
                                                                 NaN
          40.00000
                       45.000000
                                    105.000000
                                                      NaN
max
                                                                 NaN
[8 rows x 24 columns]
<IPython.core.display.HTML object>
```

Row Id 0 Row Id. 0 0 zipconvert 2 zipconvert_3 0 zipconvert_4 0 zipconvert_5 0 homeowner dummy 0 NUMCHLD 0 INCOME 0 0 gender dummy WEALTH 0 HV0 Icmed 0 Icavg 0 0 IC15 NUMPROM 0 0 RAMNTALL MAXRAMNT 0 0 LASTGIFT totalmonths 0 TIMELAG 0 AVGGIFT 0 TARGET_B 2000 TARGET_D 2000 dtype: int64

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

```
import seaborn as sns
from IPython.display import display, HTML
# Load the data
df = pd.read_csv(
   "/Users/gabrielmancillas/Desktop/ADS 505-01/Mod 04/Assignment 4.1/

→Fundraising.csv"

)
dff = pd.read_csv(
   "/Users/gabrielmancillas/Desktop/ADS 505-01/Mod 04/Assignment 4.1/
→FutureFundraising.csv"
# 3. Data Types and Conversion
# Convert 'booking_created' to datetime if necessary
if "booking_created" in df.columns:
   df["booking_created"] = pd.to_datetime(df["booking_created"])
if "booking_created" in dff.columns:
   dff["booking_created"] = pd.to_datetime(dff["booking_created"])
# 4. Descriptive Statistics
display(
   HTML(
        "<h3>Updated summary statistics for Fundraising dataset after handling...
 ⇔missing values:</h3>"
display(df.describe())
display(
   HTML(
        "<h3>Updated summary statistics for Future Fundraising dataset after
 ⇔handling missing values:</h3>"
display(dff.describe())
# 5. Data Visualization
# Distribution of numerical features
display(HTML("<h3>Distribution of numerical features in Fundraising dataset:/
⇔h3>"))
df.hist(bins=30, figsize=(20, 15))
plt.show()
display(
```

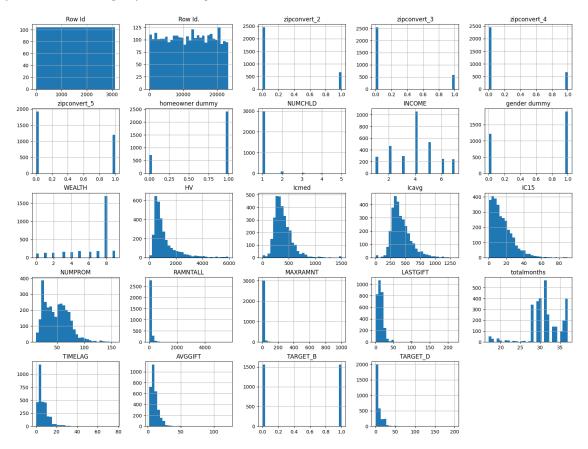
```
HTML("<h3>Distribution of numerical features in Future Fundraising dataset:
  </h3>")
)
dff.hist(bins=30, figsize=(20, 15))
plt.show()
# Correlation heatmap
display(HTML("<h3>Correlation Heatmap for Fundraising dataset:</h3>"))
plt.figure(figsize=(12, 8))
sns.heatmap(df.corr(), annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap - Fundraising")
plt.show()
display(HTML("<h3>Correlation Heatmap for Future Fundraising dataset:</h3>"))
plt.figure(figsize=(12, 8))
sns.heatmap(dff.corr(), annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap - Future Fundraising")
plt.show()
<IPython.core.display.HTML object>
            Row Id
                          Row Id.
                                   zipconvert_2
                                                  zipconvert_3
                                                                zipconvert_4
                                                   3120.000000
                                                                  3120.000000
count
       3120,000000
                     3120.000000
                                    3120.000000
                    11615.770833
       1560.500000
                                       0.214423
                                                      0.185256
                                                                     0.214423
mean
std
        900.810746
                      6698.678131
                                       0.410487
                                                      0.388568
                                                                     0.410487
          1.000000
                        17.000000
                                       0.000000
                                                      0.000000
                                                                     0.000000
min
25%
        780.750000
                     5820.750000
                                       0.000000
                                                      0.000000
                                                                     0.000000
50%
       1560.500000
                    11735.500000
                                       0.000000
                                                      0.000000
                                                                     0.000000
75%
                    17435.750000
       2340.250000
                                       0.000000
                                                      0.000000
                                                                     0.000000
max
       3120.000000
                    23293.000000
                                       1.000000
                                                      1.000000
                                                                     1.000000
       zipconvert 5 homeowner dummy
                                                                   gender dummy
                                            NUMCHLD
                                                          INCOME
        3120.000000
                          3120.000000
                                       3120.000000
                                                     3120.000000
                                                                    3120.000000
count
                             0.770192
                                                                       0.609295
mean
           0.384615
                                           1.069231
                                                        3.893910
std
           0.486582
                             0.420777
                                           0.347688
                                                        1.636186
                                                                       0.487987
min
           0.000000
                             0.000000
                                           1.000000
                                                        1.000000
                                                                       0.000000
25%
           0.000000
                             1.000000
                                           1.000000
                                                        3.000000
                                                                       0.000000
50%
           0.000000
                             1.000000
                                           1.000000
                                                        4.000000
                                                                       1.000000
75%
           1.000000
                             1.000000
                                           1.000000
                                                        5.000000
                                                                       1.000000
           1.000000
                             1.000000
                                           5.000000
                                                        7.000000
                                                                       1.000000
max
                  IC15
                            NUMPROM
                                        RAMNTALL
                                                      MAXRAMNT
                                                                    LASTGIFT
count
          3120.000000
                        3120.000000
                                     3120.000000
                                                   3120.000000
                                                                3120.000000
mean
            14.702885
                          49.089423
                                      110.399875
                                                     16.651397
                                                                   13.522917
std
            12.079882
                          22.717130
                                      147.299933
                                                     22.223521
                                                                   10.581439
min
             0.000000
                          11.000000
                                       15.000000
                                                      5.000000
                                                                    0.000000
25%
             5.000000
                          29.000000
                                       45.000000
                                                     10.000000
                                                                    7.000000
50%
            12.000000
                                       81.000000
                          48.000000
                                                     15.000000
                                                                   10.000000
```

75%	21.000000 65.000000 134.625000 20.000000 16.000000 90.000000 157.000000 5674.900000 1000.000000 219.000000									
max	90.000000 157.000000 5674.900000 1000.000000 219.000000									
	totalmonths TIMELAG AVGGIFT TARGET_B TARGET_D									
count	3120.000000 3120.000000 3120.000000 3120.000000									
	31.136859 6.861859 10.690713 0.50000 6.499612									
mean										
std	4.132952 5.561209 7.443980 0.50008 10.597849 4.7.00000 0.00000 0.00000 0.00000									
min	17.000000 0.000000 2.138889 0.00000 0.000000									
25%	29.000000 3.000000 6.356092 0.00000 0.000000									
50%	31.000000 5.000000 9.000000 0.50000 0.500000									
75%	34.000000 9.000000 12.811652 1.00000 10.000000									
max	37.000000 77.000000 122.166667 1.00000 200.000000									
[8 row	[8 rows x 24 columns]									
<ipyth< td=""><td>on.core.display.HTML object></td></ipyth<>	on.core.display.HTML object>									
	Row Id Row Id. zipconvert_2 zipconvert_3 zipconvert_4 \									
count	2000.000000 2000.000000 2000.000000 2000.000000 2000.000000									
mean	1000.500000 3.482500 0.237500 0.168500 0.230000									
std	577.494589 1.300592 0.425658 0.374403 0.420938									
min	1.000000 1.000000 0.000000 0.000000 0.000000									
25%	500.750000 2.000000 0.000000 0.000000 0.000000									
50%										
75%	1500.250000 5.000000 0.000000 0.000000 0.000000									
max	2000.000000 5.000000 1.000000 1.000000 1.000000									
	zipconvert_5 homeowner dummy NUMCHLD INCOME gender dummy \									
count	2000.00000 2000.00000 2000.000000 2000.000000 2000.000000									
mean	0.30450 0.77850 1.052000 3.821500 0.595500									
std	0.46031 0.41536 0.278091 1.638591 0.490918									
min	0.00000 0.00000 1.000000 0.000000									
25%	0.00000 1.00000 1.000000 2.750000 0.000000									
50%	0.00000 1.00000 1.000000 1.000000									
75%	1.00000 1.00000 5.000000 1.000000									
max	1.00000 1.00000 5.000000 7.000000 1.000000									
	IC15 NUMPROM RAMNTALL MAXRAMNT LASTGIFT \									
count	2000 000000 2000 000000 2000 000000 2000 000000									
mean										
std	12.577988 22.963991 98.844943 11.752492 9.714386									
min	0.000000 4.000000 15.000000 5.000000 0.000000									
25%	5.750000 27.000000 42.000000 14.000000 10.000000									
50%	12.000000 47.000000 80.000000 17.000000 15.000000									
75%	22.000000 64.000000 132.000000 22.000000 20.000000									
max	87.000000 148.000000 1526.000000 200.000000 105.000000									
	+.+-l									
	totalmonths TIMELAG AVGGIFT TARGET_B TARGET_D									
count	2000.00000 2000.000000 2000.000000 0.0 0.									

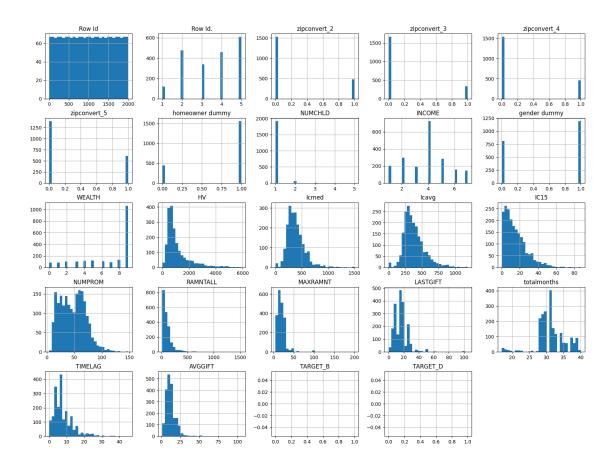
mean	31.29300	7.678000	13.051707	NaN	NaN
std	4.04244	5.652459	7.982889	NaN	NaN
min	17.00000	0.000000	1.636364	NaN	NaN
25%	29.00000	4.000000	8.408824	NaN	NaN
50%	31.00000	6.000000	11.600000	NaN	NaN
75%	33.00000	10.000000	15.211806	NaN	NaN
max	40.00000	45.000000	105.000000	NaN	NaN

[8 rows x 24 columns]

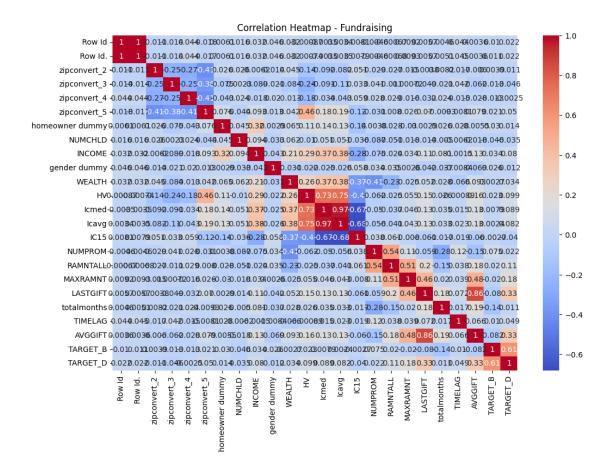
<IPython.core.display.HTML object>



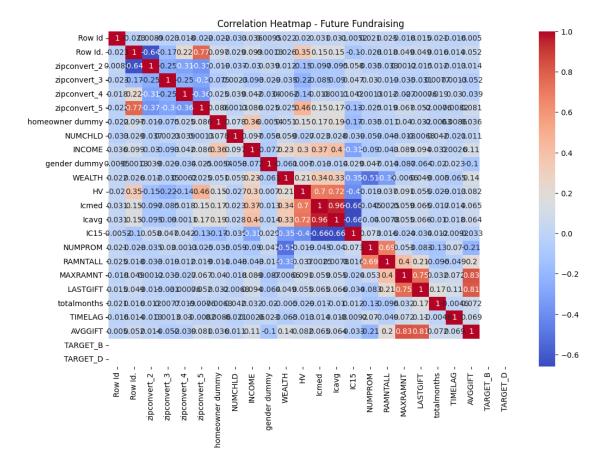
<IPython.core.display.HTML object>



<IPython.core.display.HTML object>



<IPython.core.display.HTML object>



Question 1

```
[4]: from sklearn.model_selection import train_test_split
    from sklearn.linear_model import LinearRegression
    from sklearn.metrics import mean_squared_error
    from sklearn.linear_model import LogisticRegression
    from sklearn.ensemble import RandomForestClassifier
    from sklearn.metrics import classification_report, confusion_matrix

X = df.drop("TARGET_B", axis=1)
    y = df["TARGET_B"]

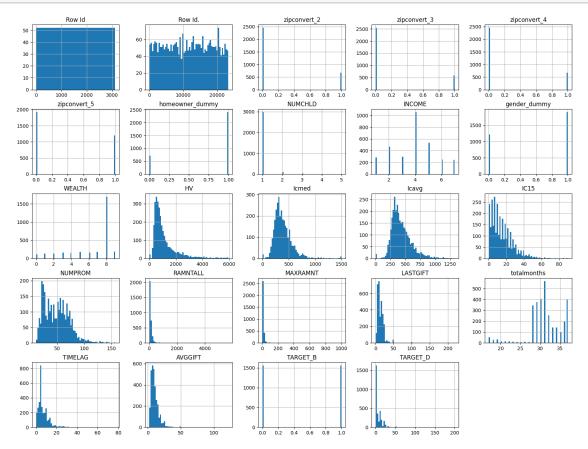
# 6. Data Preparation
    train, val = train_test_split(df, test_size=0.2, random_state=12345)
```

```
[5]: df.duplicated().sum()
```

[5]: 0

```
[6]: df.rename(columns={"gender dummy": "gender_dummy"}, inplace=True)
df.rename(columns={"homeowner dummy": "homeowner_dummy"}, inplace=True)
```

```
[7]: df.hist(bins=60, figsize=(20, 15)) plt.show()
```



```
[8]: from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
df_num = df.copy()
l_cat = []

for col in df:
    if df[col].nunique() <= 21:
        df_num = df_num.drop(col, axis=1)
        l_cat.append(col)

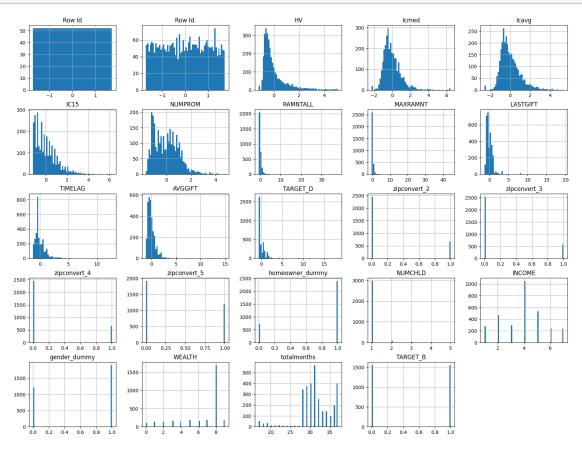
# categorical
df_cat = df.filter(l_cat, axis=1)</pre>
```

```
# Ratio
     df_num_scaled = scaler.fit_transform(df_num)
     df_num.head()
[8]:
       Row Id Row Id.
                                      Icavg
                                            IC15
                                                   NUMPROM
                                                            RAMNTALL MAXRAMNT
                           HV
                               Icmed
             1
                     17
                         1399
                                 637
                                        703
                                                1
                                                        74
                                                                102.0
                                                                            6.0
             2
     1
                                                4
                     25
                          698
                                 422
                                        463
                                                        46
                                                                94.0
                                                                           12.0
     2
             3
                     29
                          828
                                 358
                                        376
                                               13
                                                        32
                                                                30.0
                                                                           10.0
     3
             4
                                        546
                                                4
                                                        94
                                                               177.0
                     38
                        1471
                                 484
                                                                           10.0
             5
     4
                     40
                          547
                                 386
                                        432
                                                7
                                                        20
                                                                23.0
                                                                           11.0
       LASTGIFT
                 TIMELAG
                            AVGGIFT
                                     TARGET_D
     0
             5.0
                        3
                          4.857143
                                          5.0
                                         10.0
     1
            12.0
                        6 9.400000
     2
             5.0
                        7 4.285714
                                          5.0
     3
             8.0
                        3 7.080000
                                          0.0
     4
            11.0
                        6 7.666667
                                          0.0
[9]: # concatenating datasets for visualization purposes
     df_num_scaled = pd.DataFrame(df_num_scaled, columns=df_num.columns)
     Scaled_df = pd.concat([df_num_scaled, df_cat], axis=1)
     Scaled_df.head()
[9]:
          Row Id
                   Row Id.
                                  HV
                                         Icmed
                                                              IC15
                                                                     NUMPROM \
                                                   Icavg
     0 -1.731496 -1.731779 0.272204
                                      1.439813 1.610958 -1.134538 1.096731
     1 -1.730385 -1.730585 -0.468427
                                      2 -1.729275 -1.729987 -0.331078 -0.174881 -0.333524 -0.140991 -0.752391
     3 -1.728165 -1.728644 0.348274 0.554336
                                                0.677369 -0.886151 1.977265
     4 -1.727055 -1.728345 -0.627964 -0.012833 -0.000524 -0.637764 -1.280711
       RAMNTALL MAXRAMNT LASTGIFT
                                         zipconvert_3
                                                       zipconvert_4
     0 -0.057035 -0.479362 -0.805588
                                                                  0
                                                    1
                                                    0
                                                                  0
     1 -0.111354 -0.209334 -0.143946
     2 -0.545912 -0.299343 -0.805588
                                                    0
                                                                  0
     3 0.452212 -0.299343 -0.522027
                                                    0
                                                                  0
     4 -0.593441 -0.254339 -0.238467
                                                    1
                                                                  0
       zipconvert_5
                     homeowner_dummy
                                       NUMCHLD
                                                INCOME
                                                        gender_dummy
                                                                      WEALTH
     0
                   0
                                    1
                                             1
                                                     5
                                                                            9
                                                                    1
     1
                   0
                                    1
                                             1
                                                     1
                                                                   0
                                                                            7
     2
                                    0
                                             2
                                                     5
                   1
                                                                    1
                                                                            8
                                    1
                                                     3
                                                                   0
     3
                   1
                                             1
                                                                            4
     4
                   0
                                    1
                                             1
                                                     4
                                                                    0
                                                                            8
       totalmonths
                    TARGET_B
     0
                 29
                            1
```

```
1 34 1
2 29 1
3 30 0
4 30 0
```

[5 rows x 24 columns]

```
[10]: Scaled_df.hist(bins=60, figsize=(20, 15))
plt.show()
```



```
[11]: corr = Scaled_df.corr()
  plt.figure(figsize=(20, 20))
  sns.heatmap(
      corr, cbar=True, fmt=".2f", annot=True, annot_kws={"size": 15}, cmap="Blues"
)
```

[11]: <Axes: >

```
HV -0.00-0.00 1.00 0.73 0.75 -0.40-0.06-0.02 0.06 0.15 -0.00 0.16 0.10-0.14-0.24-0.18 0.46 0.11-0.01 0.29 -0.02 0.26 0.03 0.02
                               kmed -0.00-0.00 0.73 1.00 0.97 -0.67-0.05-0.04 0.05 0.13 0.02 0.13 0.09-0.09-0.09-0.03 0.18 0.14 0.05 0.37 -0.03 0.37 0.03 0.01
                                 κανς -0.00 0.00 0.75 0.97 1.00 -0.68-0.06-0.04 0.04 0.13 0.02 0.13 0.08-0.08-0.11-0.04 0.19 0.13 0.05 0.38 -0.03 0.38 0.03 0.00
                                   K15 - 0.01 0.01 -0.40-0.67-0.68 1.00 0.04 0.06 -0.01-0.06-0.02-0.06-0.040.05 0.03 0.06 -0.12-0.14-0.04-0.28 0.06 -0.37-0.02-0.00
                      NUMPROM -0.00-0.00-0.06-0.05-0.060.04 1.00 0.54 0.11-0.060.12-0.150.02-0.030.04 0.03-0.030.00-0.09-0.070.03-0.41-0.280.07
                      RAMNTALL -0.00-0.00-0.02-0.04-0.040.06 0.54 1.00 0.51 0.20 0.04 0.18 0.11-0.03-0.010.03 0.01-0.03-0.05-0.02 0.04 -0.23-0.15 0.02
                   MAXRAMNT -0.01 0.01 0.06 0.05 0.04-0.010.11 0.51 1.00 0.46 0.04 0.48 0.18-0.020.00-0.020.03-0.03-0.020.03 0.00-0.020.02-0.02
                       LASTGIFT -0.01 0.01 0.15 0.13 0.13 -0.06-0.06 0.20 0.46 1.00 0.07 0.86 0.33 -0.00-0.05-0.030.07 0.00 -0.010.11 -0.040.05 0.18 -0.08
                         TIMELAG -0.04 0.04-0.000.02 0.02-0.02 0.12 0.04 0.04 0.07 1.00 0.07 0.05-0.02 0.04 -0.02-0.01 0.03-0.01-0.00-0.01-0.070.02 0.01
                         AVGGIFT -0.00 0.00 0.16 0.13 0.13 -0.06-0.15 0.18 0.48 0.86 0.07 1.00 0.33 -0.01-0.06-0.030.08 -0.01-0.020.13 -0.07 0.09 0.19 -0.08
                      TARGET_D -0.02 0.02 0.10 0.09 0.08 -0.040.02 0.11 0.18 0.33 0.05 0.33 1.00 -0.01-0.05-0.000.05 0.01 -0.030.08 -0.010.03 0.01 0.61
                \underline{\textbf{zipconvert}}. \underline{\textbf{z}} - 0.01 - 0.01 - 0.01 + 0.09 - 0.080.05 - 0.03 - 0.03 - 0.02 - 0.00 - 0.02 - 0.01 - 0.01 \\ \underline{\textbf{1}.00} - 0.25 - 0.27 - 0.41 \\ \underline{\textbf{0}.03} \ 0.03 - 0.01 - 0.01 \\ \underline{\textbf{0}.00} - 0.01 - 0.01 \\ \underline{\textbf{0}.00} - 0.02 - 0.01 \\ \underline{\textbf{0}.00} - 0.01 \\ \underline{\textbf{0
               {\tt zpconvert\_3} - 0.01 - 0.01 - 0.024 - 0.09 + 0.11 \ 0.03 \ 0.04 - 0.010.00 - 0.055.004 - 0.06 - 0.05 \cdot 0.25 \ 1.00 - 0.25 \cdot 0.38 - 0.07 \ 0.00 - 0.09 \cdot 0.02 - 0.08 \cdot 0.02 - 0.01 \ 0.00 - 0.00 - 0.00 - 0.00 \ 0.00 - 0.00 - 0.00 \ 0.00 - 0.00 - 0.00 \ 0.00 - 0.00 - 0.00 \ 0.00 - 0.00 - 0.00 \ 0.00 - 0.00 \ 0.00 - 0.00 - 0.00 \ 0.00 - 0.00 \ 0.00 - 0.00 \ 0.00 - 0.00 \ 0.00 - 0.00 \ 0.00 - 0.00 \ 0.00 - 0.00 \ 0.00 - 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 
               zipconvert_5 -0.02-0.02 0.46 0.18 0.19 -0.12-0.030.01 0.03 0.07 -0.010.08 0.05 -0.41-0.38-0.41 1.00 0.08 -0.040.09 0.01 0.04-0.010.02
homeowner_dummy -0.01 0.01 0.11 0.14 0.13 -0.140.00 -0.03-0.030.00 0.03 -0.010.01 0.03 -0.07-0.040.08 1.00 0.05 0.32 0.00 0.06 0.03 0.03
                      NUMCHLD -0.02 0.02 -0.01 0.05 0.05 -0.04-0.09-0.05-0.02-0.01-0.01-0.02-0.030.03 0.00 0.02-0.040.05 1.00 0.09-0.030.06-0.01-0.05
                           NCOME -0.03 0.03 0.29 0.37 0.38 -0.28-0.07-0.020.03 0.11-0.000.13 0.08-0.01-0.09-0.020.09 0.32 0.09 1.00-0.04 0.21 0.08 0.03
         \frac{1}{2} \frac{1}
                            WEALTH - 0.03 0.03 0.26 0.37 0.38 -0.37-0.41-0.23-0.02 0.05 -0.07 0.09 0.03 0.04 -0.08-0.010.04 0.06 0.06 0.21 -0.03 1.00 0.03 0.00
                 totalmonths -0.00 0.01 0.03 0.03 0.03 -0.02-0.28-0.15 0.02 0.18 0.02 0.19 0.01 0.01 -0.02 0.02-0.01 0.03 -0.01 0.08 -0.04 0.03 1.00 -0.14
                      TARGET_B -0.01 0.01 0.02 0.01 0.00 -0.000.07 0.02 -0.02-0.080.01 -0.08 0.61 0.00 -0.01-0.010.02 0.03 -0.050.03 0.03 0.00 -0.14 1.00
```

```
[12]: from sklearn.pipeline import Pipeline
    from sklearn.preprocessing import StandardScaler
    from sklearn.impute import SimpleImputer
    from sklearn.base import BaseEstimator, TransformerMixin
    from sklearn.compose import ColumnTransformer
[13]: Row_ix, RowId_ix, Target_d_ix = 0, 1, -1

class columnDropperTransformer(BaseEstimator, TransformerMixin):
    def __init__(self, columns):
```

```
self.columns = columns
          def transform(self, X):
              return X.drop(X.columns[self.columns], axis=1)
          def fit(self, X, y=None):
             return self
      num_pipeline = Pipeline(
          Γ
              ("drop_data", columnDropperTransformer([Row_ix, RowId_ix,_
       →Target_d_ix])),
              ("imputer", SimpleImputer(strategy="median")),
              ("std_scaler", StandardScaler()),
          ]
      )
      num_attribs = list(df.drop(df_cat.columns, axis=1))
      cat_attribs = list(df_cat.drop(["TARGET_B"], axis=1).columns)
      full_pipeline = ColumnTransformer(
          ("num", num_pipeline, num_attribs),
          ],
          remainder="passthrough",
      )
      X_train, X_test, y_train, y_test = train_test_split(
          df.drop("TARGET_B", axis=1), df["TARGET_B"], test_size=0.2,_
       →random_state=12345
      X_prepared = full_pipeline.fit_transform(X_train)
[14]: X_prepared = pd.DataFrame(X_prepared, columns=X_train.columns[2:-1])
[15]: X_prepared.head()
[15]:
        zipconvert_2 zipconvert_3 zipconvert_4 zipconvert_5 homeowner_dummy \
           -0.162999
                          -0.140958
                                                      -0.643554
                                                                       -1.069037
      0
                                        -0.153803
      1
            -0.793101
                          -0.625512
                                        -0.587747
                                                       0.172578
                                                                       -0.673346
      2
           -0.649169
                          -0.777299
                                                                       -1.244899
                                        -0.938470
                                                       0.825483
            -0.042523
                          1.236807
                                         1.231252
                                                      -1.214845
                                                                        0.293897
            -0.728065
                          -0.888221
                                        -1.081136
                                                      -0.561940
                                                                       -0.189725
          NUMCHLD
                    INCOME gender_dummy
                                                           HV Icmed Icavg IC15 \
                                             WEALTH
                                -0.697739 -1.066196 -0.809061
      0 -0.501734 -0.443354
                                                                 0.0
                                                                        0.0
                                                                              0.0
```

```
1 -0.342814 0.339144
                                 1.060521 -0.884070 0.095686
                                                                 0.0
                                                                        0.0
                                                                              1.0
      2 -0.482663 0.133224
                                 0.135121 -0.519818 0.903901
                                                                 1.0
                                                                        0.0
                                                                              0.0
      3 0.477211 3.427954
                                -0.790279 -0.519818 1.049645
                                                                 0.0
                                                                        0.0
                                                                              1.0
                                 0.135121 6.765225 -0.089806
      4 -0.387312 -0.072697
                                                                 0.0
                                                                        0.0
                                                                              1.0
        NUMPROM RAMNTALL MAXRAMNT LASTGIFT totalmonths TIMELAG AVGGIFT
     0
             1.0
                      0.0
                                 2.0
                                           3.0
                                                        0.0
                                                                 8.0
                                                                         29.0
             0.0
                      1.0
                                 1.0
                                           4.0
                                                                 8.0
      1
                                                        0.0
                                                                         37.0
      2
            0.0
                                           5.0
                                                        1.0
                      1.0
                                 1.0
                                                                 8.0
                                                                         37.0
      3
             0.0
                      1.0
                                 1.0
                                           4.0
                                                        1.0
                                                                 8.0
                                                                         26.0
            0.0
                                 2.0
      4
                       1.0
                                           3.0
                                                        1.0
                                                                 3.0
                                                                         29.0
[16]: def score(y_train, train_pred, y_test, test_pred):
          from sklearn import metrics
          print("Training precision: ", metrics.precision_score(y_train, train_pred))
          print("Validation precision: ", metrics.precision_score(y_test, test_pred))
          print("\n")
          print("Training accuracy: ", metrics.accuracy_score(y_train, train_pred))
          print("Validation accuracy: ", metrics.accuracy_score(y_test, test_pred))
      def net(train, test):
          n_of_1s = np.unique(train, return_counts=True)
          n2_of_1s = np.unique(test, return_counts=True)
          n = n_of_1s[1][1]
          n2 = n2_of_1s[1][1]
          net = ((n * 13) - (0.68 * n)) / 9.8
          net2 = ((n2 * 13) - (0.68 * n2)) / 9.8
          return net, net2
      def net profit(
          train_pred_1, test_pred_1, train_pred_2, test_pred_2, train_pred_3,__
       →test pred 3
      ):
          import matplotlib.pyplot as plt
          net1, net2 = net(train_pred_1, test_pred_1)
          net3, net4 = net(train_pred_2, test_pred_2)
          net5, net6 = net(train pred 3, test pred 3)
          net df = pd.DataFrame(
              [[net1, net3, net5], [net2, net4, net6]],
              columns=["LR", "XGB", "SVC"],
             index=["train", "test"],
          plt.bar(["LR", "XGB", "SVC"], [net2, net4, net6])
```

```
return net_df
[17]: X_test_pre = full_pipeline.transform(X_test)
      X_test_pre = pd.DataFrame(X_test_pre, columns=X_train.columns[2:-1])
[18]: # remove an error message
      import warnings
      warnings.filterwarnings("ignore")
      from sklearn.linear_model import LogisticRegression
      m = LogisticRegression().fit(X_prepared, y_train)
      train_pred = m.predict(X_prepared)
      test_pred = m.predict(X_test_pre)
      score(y_train, train_pred, y_test, test_pred)
     Training precision: 0.5737082066869301
     Validation precision: 0.5391566265060241
     Training accuracy: 0.572916666666666
     Validation accuracy: 0.5608974358974359
[19]: import xgboost as xgb
      from xgboost import XGBClassifier
      from sklearn.metrics import accuracy_score
      from sklearn.model_selection import RandomizedSearchCV
      xgb = XGBClassifier()
      xgb.fit(X_prepared, y_train)
      train_pred = xgb.predict(X_prepared)
      test_pred = xgb.predict(X_test_pre)
      score(y_train, train_pred, y_test, test_pred)
      param_dist = {
          "n_estimators": [100, 200, 300],
          "learning_rate": [0.01, 0.1, 0.2],
          "max_depth": [3, 4, 5],
      }
      clf = RandomizedSearchCV(
```

xgb,

```
param_distributions=param_dist,
    cv=5,
    n_iter=5,
    scoring="accuracy",
    n_jobs=-1,
    verbose=3,
    random_state=123,
)
clf.fit(X_prepared, y_train)
```

Training precision: 0.9992063492063492 Validation precision: 0.49693251533742333

```
Training accuracy: 0.999198717948718
Validation accuracy: 0.5160256410256411
Fitting 5 folds for each of 5 candidates, totalling 25 fits
[CV 2/5] END learning_rate=0.2, max_depth=4, n_estimators=300;, score=0.505
total time=
[CV 3/5] END learning_rate=0.2, max_depth=4, n_estimators=300;, score=0.503
total time=
             0.1s
[CV 5/5] END learning_rate=0.2, max_depth=4, n_estimators=300;, score=0.513
total time=
             0.1s
[CV 1/5] END learning_rate=0.2, max_depth=4, n_estimators=300;, score=0.528
total time=
[CV 4/5] END learning_rate=0.2, max_depth=4, n_estimators=300;, score=0.487
total time=
[CV 2/5] END learning_rate=0.01, max_depth=4, n_estimators=300;, score=0.579
total time=
             0.1s
[CV 1/5] END learning_rate=0.01, max_depth=5, n_estimators=200;, score=0.544
total time=
              0.1s
[CV 2/5] END learning rate=0.01, max_depth=5, n_estimators=200;, score=0.555
total time=
              0.1s
[CV 5/5] END learning_rate=0.01, max_depth=5, n_estimators=200;, score=0.543
total time=
[CV 4/5] END learning_rate=0.01, max_depth=5, n_estimators=200;, score=0.535
total time=
              0.1s
[CV 3/5] END learning_rate=0.01, max_depth=5, n_estimators=200;, score=0.547
total time=
              0.1s
[CV 3/5] END learning_rate=0.01, max_depth=4, n_estimators=300;, score=0.545
total time=
[CV 4/5] END learning_rate=0.01, max_depth=4, n_estimators=300;, score=0.525
total time=
[CV 1/5] END learning_rate=0.01, max_depth=4, n_estimators=300;, score=0.552
total time=
[CV 1/5] END learning_rate=0.2, max_depth=5, n_estimators=100;, score=0.532
total time=
              0.1s
[CV 2/5] END learning_rate=0.2, max_depth=5, n_estimators=100;, score=0.519
```

```
total time=
                   0.1s
     [CV 1/5] END learning_rate=0.01, max_depth=5, n_estimators=300;, score=0.546
     total time=
                   0.2s
     [CV 3/5] END learning_rate=0.2, max_depth=5, n_estimators=100;, score=0.501
     total time=
     [CV 5/5] END learning_rate=0.01, max_depth=4, n_estimators=300;, score=0.521
     total time=
     [CV 4/5] END learning_rate=0.2, max_depth=5, n_estimators=100;, score=0.495
     total time=
     [CV 5/5] END learning_rate=0.2, max_depth=5, n_estimators=100;, score=0.535
     total time=
     [CV 3/5] END learning_rate=0.01, max_depth=5, n_estimators=300;, score=0.529
     total time=
     [CV 5/5] END learning_rate=0.01, max_depth=5, n_estimators=300;, score=0.539
     total time=
     [CV 2/5] END learning rate=0.01, max_depth=5, n_estimators=300;, score=0.553
     total time=
                   0.2s
     [CV 4/5] END learning rate=0.01, max_depth=5, n_estimators=300;, score=0.527
     total time=
[19]: RandomizedSearchCV(cv=5,
                         estimator=XGBClassifier(base score=None, booster=None,
                                                  callbacks=None,
                                                  colsample_bylevel=None,
                                                  colsample_bynode=None,
                                                  colsample_bytree=None, device=None,
                                                  early_stopping_rounds=None,
                                                  enable_categorical=False,
                                                  eval_metric=None, feature_types=None,
                                                  gamma=None, grow_policy=None,
                                                  importance_type=None,
                                                  interaction_constraints=None,
                                                 learning rate...
                                                 max_delta_step=None, max_depth=None,
                                                 max leaves=None,
                                                 min_child_weight=None, missing=nan,
                                                 monotone constraints=None,
                                                 multi_strategy=None,
                                                 n_estimators=None, n_jobs=None,
                                                 num_parallel_tree=None,
                                                 random_state=None, ...),
                         n_iter=5, n_jobs=-1,
                         param_distributions={'learning_rate': [0.01, 0.1, 0.2],
                                               'max_depth': [3, 4, 5],
                                               'n_estimators': [100, 200, 300]},
```

random_state=123, scoring='accuracy', verbose=3)

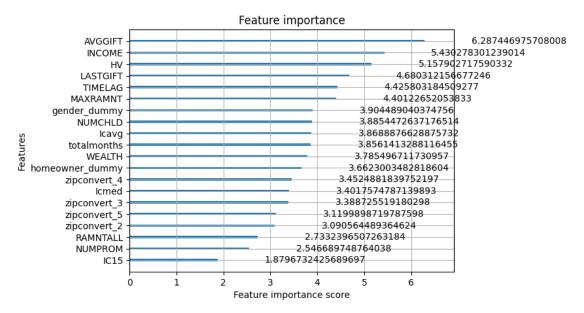
```
[20]: # optimal hyperparameters
    clf.best_params_

[20]: {'n_estimators': 200, 'max_depth': 5, 'learning_rate': 0.01}

[21]: train_pred2 = clf.predict(X_prepared)
    test_pred2 = clf.predict(X_test_pre)

[22]: import xgboost as xgb

# Plot feature importance
    xgb.plot_importance(
        clf.best_estimator_, importance_type="gain", xlabel="Feature importance_uscore"
    )
    plt.show()
```



```
[23]: score(y_train, train_pred2, y_test, test_pred2)
```

Training precision: 0.7288135593220338 Validation precision: 0.55333333333333333

Training accuracy: 0.7115384615384616 Validation accuracy: 0.5705128205128205

```
[24]: from sklearn.svm import SVC
```

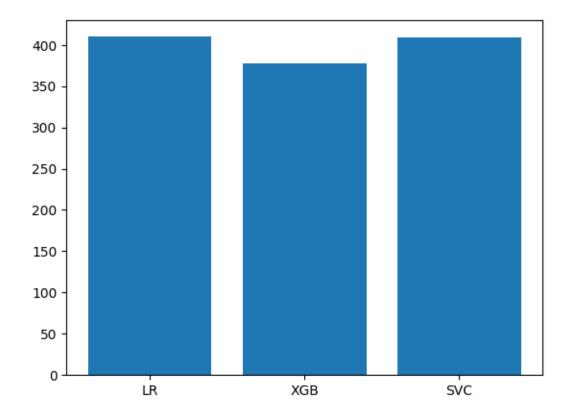
```
clf = SVC(gamma="auto", probability=True).fit(X_prepared, y_train)
train_pred3 = clf.predict(X_prepared)
test_pred3 = clf.predict(X_test_pre)
score(y_train, train_pred3, y_test, test_pred3)
```

Training precision: 0.6549570647931303 Validation precision: 0.5292307692307693

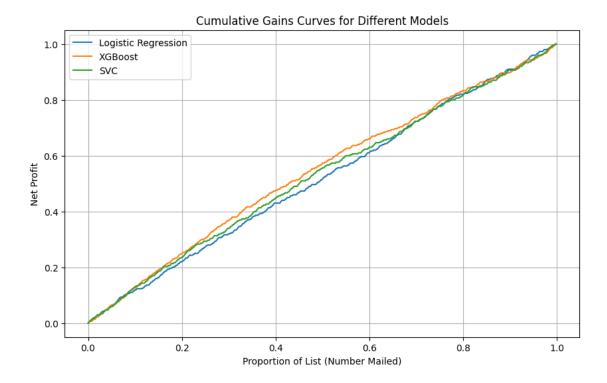
Training accuracy: 0.6542467948717948 Validation accuracy: 0.5496794871794872

Question 2.3

[25]: LR XGB SVC train 1584.000000 1483.428571 1610.400000 test 409.828571 377.142857 408.571429

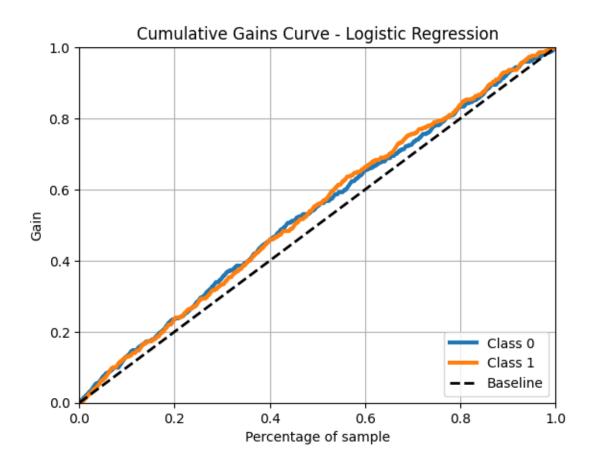


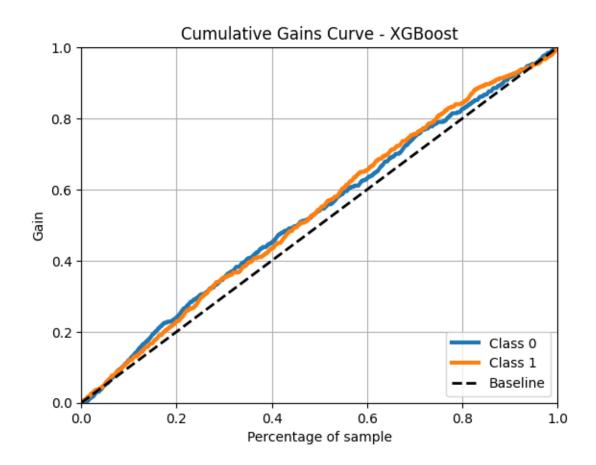
```
[26]: import numpy as np
      import matplotlib.pyplot as plt
      # Function to calculate cumulative gains
      def cumulative_gains(y_true, y_pred):
          sorted_indices = np.argsort(y_pred)[::-1]
          y_true_sorted = y_true.iloc[sorted_indices]
          gains = np.cumsum(y_true_sorted) / np.sum(y_true_sorted)
          return gains
      # Calculate cumulative gains for each model
      gains_lr = cumulative_gains(y_test, test_pred)
      gains_xgb = cumulative_gains(y_test, test_pred2)
      gains_svc = cumulative_gains(y_test, test_pred3)
      # Plot cumulative gains curves
      plt.figure(figsize=(10, 6))
      plt.plot(
          np.arange(len(gains_lr)) / len(gains_lr), gains_lr, label="Logisticu
       →Regression"
      plt.plot(np.arange(len(gains_xgb)) / len(gains_xgb), gains_xgb, label="XGBoost")
      plt.plot(np.arange(len(gains_svc)) / len(gains_svc), gains_svc, label="SVC")
      plt.xlabel("Proportion of List (Number Mailed)")
      plt.ylabel("Net Profit")
      plt.title("Cumulative Gains Curves for Different Models")
      plt.legend()
      plt.grid(True)
      plt.show()
```

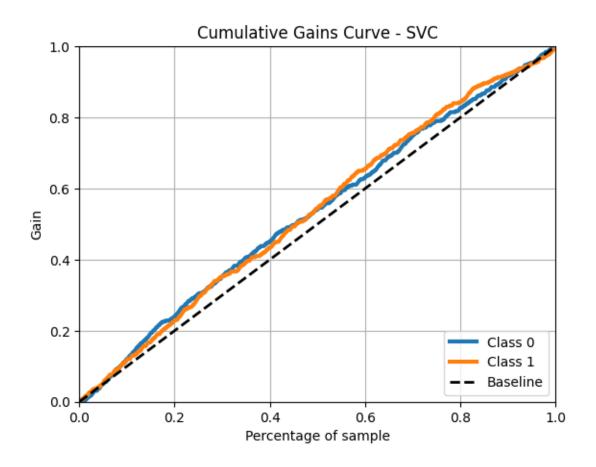


```
[27]: import scikitplot as skplt
      # Deriving class probabilities for each model
      predicted_probabilities_lr = m.predict_proba(X_test_pre)
      predicted_probabilities_xgb = clf.predict_proba(X_test_pre)
      predicted_probabilities_svc = clf.predict_proba(X_test_pre)
      # Creating the plot
      plt.figure(figsize=(10, 6))
      skplt.metrics.plot_cumulative_gain(
          y_test,
          predicted_probabilities_lr,
          title="Cumulative Gains Curve - Logistic Regression",
      skplt.metrics.plot_cumulative_gain(
          y_test, predicted_probabilities_xgb, title="Cumulative Gains Curve -_
       ⇔XGBoost"
      skplt.metrics.plot_cumulative_gain(
          y_test, predicted_probabilities_svc, title="Cumulative Gains Curve - SVC"
      plt.show()
```

<Figure size 1000x600 with 0 Axes>







[28]: # use the future fundraising dataset to predict the target variable dff.head()												
[28]:		Row Id I	Row Id.	zipconve	rt_2 z	:ipconvert_	.3 zi	ipcon	vert_4	zipconv	ert_5	\
	0	1	3		0		1		0		0	
	1	2	4		0		0		1		0	
	2	3	5		0		0		0		1	
	3	4	1		0		0		0		0	
	4	5	4		0		0		1		0	
	0	homeowne	r dummy	NUMCHLD	INCOME	gender d	lummy	•••	IC15 NU	JMPROM '	\	
	1		0	1	1		1	•••	4	21		
	2		0	1	1	•	0	•••	10	61		
	3		1	1	4	L	0	•••	21	32		
	4		1	1	7	•	1	•••	1	47		
	-		Τ.	1	'		1	•••	1	-11		
		RAMNTALL	MAXRAM	NT LASTG	IFT to	talmonths	TIMELAG		AVGGI	FT TAR	GET_B	\
	0	92.0	29	.0 1	5.0	17 8		8	15.3333	333	NaN	
	1	30.0	20	.0 2	0.0	33		9	15.0000	000	NaN	

```
220.0
                       35.0
                                 25.0
     2
                                                31
                                                          9 24.44444
                                                                              NaN
      3
             41.0
                       19.0
                                 19.0
                                                31
                                                         13 13.666667
                                                                              NaN
      4
             46.0
                       10.0
                                 10.0
                                                28
                                                          8
                                                              5.750000
                                                                              NaN
         TARGET_D
      0
              NaN
      1
              NaN
      2
              NaN
      3
              NaN
              NaN
      [5 rows x 24 columns]
[29]: # Split data
      X = dff.drop("TARGET_B", axis=1)
      y = dff["TARGET_B"]
[30]: import pandas as pd
      # Ensure X has all necessary columns
      expected_columns = ["homeowner_dummy", "gender_dummy"] # Add all expected_
       ⇔columns here
      for col in expected_columns:
          if col not in X.columns:
              X[col] = 0 # or any default value that makes sense
      # Process data using full pipeline
      X_prepared = full_pipeline.transform(X)
      # Predict target variable
      from sklearn.svm import SVC
      future_pred = clf.predict(X_prepared)
      result_df = pd.DataFrame(future_pred, columns=["TARGET_B"])
[31]: # predict probability
      future_prob = clf.predict_proba(X_prepared)
      future_prob = future_prob.reshape(2, 2000)
      result_df["prop_0"] = future_prob[0]
      result_df["prop_1"] = future_prob[1]
[32]: result_df.sort_values(["prop_1", "prop_0"], ascending=[False, True],
       →ignore_index=True)
```

```
[32]:
           TARGET_B
                      prop_0
                                prop_1
                  0 0.410650 0.631504
     0
     1
                  1 0.500000 0.630064
     2
                  0 0.391967 0.629888
     3
                  1 0.563053 0.629867
     4
                  0 0.514519 0.624055
                  0 0.485481 0.375945
     1995
     1996
                  1 0.436947 0.370133
     1997
                  0 0.608033 0.370112
     1998
                  1 0.500000 0.369936
     1999
                  0 0.589350 0.368496
```

[2000 rows x 3 columns]

```
[33]: len(result_df[result_df.prop_1 >= 0.53])
```

[33]: 633

```
[34]: pip install scipy==1.11.4
```

Requirement already satisfied: scipy==1.11.4 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (1.11.4)

Requirement already satisfied: numpy<1.28.0,>=1.21.6 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from scipy==1.11.4) (1.26.4)

Note: you may need to restart the kernel to use updated packages.