In [123]:
```
# Import libraries that we will need for the initial steps.

import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
%matplotlib inline

# import warnings
# warnings.filterwarnings("ignore")
```

In [124]:
```
# Import the datasets for the training and testing. Turn them into dataframes.
trainDF = pd.read_csv("Dataset for the project/Dataset for the project/train.csv")
testDF = pd.read_csv("Dataset for the project/Dataset for the project/test.csv")
```

Determine the output variable

In [125]:
```
trainDF.head()
```

Out[125]:

	Id	v2a1	hacdor	rooms	hacapo	v14a	refrig	v18q	v18q1	r4h1	...	SQBescolari	SQBage	SQBhogar
0	ID_279628684	190000.0	0	3	0	1	1	0	NaN	0	...	100	1849	
1	ID_f29eb3ddd	135000.0	0	4	0	1	1	1	1.0	0	...	144	4489	
2	ID_68de51c94	NaN	0	8	0	1	1	0	NaN	0	...	121	8464	
3	ID_d671db89c	180000.0	0	5	0	1	1	1	1.0	0	...	81	289	
4	ID_d56d6f5f5	180000.0	0	5	0	1	1	1	1.0	0	...	121	1369	

5 rows × 143 columns

In [126]:
```
testDF.head()
```

Out[126]:

	Id	v2a1	hacdor	rooms	hacapo	v14a	refrig	v18q	v18q1	r4h1	...	age	SQBescolari	SQBage	SQB
0	ID_2f6873615	NaN	0	5	0	1	1	0	NaN	1	...	4	0	16	
1	ID_1c78846d2	NaN	0	5	0	1	1	0	NaN	1	...	41	256	1681	
2	ID_e5442cf6a	NaN	0	5	0	1	1	0	NaN	1	...	41	289	1681	
3	ID_a8db26a79	NaN	0	14	0	1	1	1	1.0	0	...	59	256	3481	
4	ID_a62966799	175000.0	0	4	0	1	1	1	1.0	0	...	18	121	324	

5 rows × 142 columns

Is idhogar the output variable? Check the column's properties

In [127]:
```
trainDF["idhogar"].unique()
```

Out[127]:
```
array(['21eb7fcc1', '0e5d7a658', '2c7317ea8', ..., 'a8eeafc29',
       '212db6f6c', 'd6c086aa3'], dtype=object)
```

In [128]:
```
len(trainDF["idhogar"].unique())
```

Out[128]:
```
2988
```

In [129]:
```
# Maybe the info() function can tell me something important
trainDF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9557 entries, 0 to 9556
Columns: 143 entries, Id to Target
dtypes: float64(8), int64(130), object(5)
memory usage: 10.4+ MB
```

In [130]:
```
# If I look at all the values in a large table, I might notice patterns. The "Target" column looks
 like something worth investigating.
#trainDF.iloc[150:200]
trainDF.iloc[1086:1100]
```

Out[130]:

	Id	v2a1	hacdor	rooms	hacapo	v14a	refrig	v18q	v18q1	r4h1	...	SQBescolari	SQBage	SQBhc
1086	ID_30d6500c3	150000.0	0	4	0	1	1	1	1.0	1	...	81	3969	
1087	ID_4203fd5e2	NaN	0	3	0	1	1	0	NaN	0	...	49	5929	
1088	ID_7c33c1884	NaN	0	6	0	1	1	0	NaN	0	...	256	3600	
1089	ID_77cce5b75	NaN	0	4	0	1	1	0	NaN	0	...	9	4356	
1090	ID_df05b9e45	NaN	0	4	0	1	1	0	NaN	0	...	4	4489	
1091	ID_3490fd704	NaN	0	7	0	1	1	0	NaN	0	...	121	400	
1092	ID_d82b1a426	NaN	0	7	0	1	1	0	NaN	0	...	81	2704	
1093	ID_9d6957e8a	NaN	0	7	0	1	1	0	NaN	0	...	121	484	
1094	ID_5038a636a	160000.0	0	5	0	1	1	1	1.0	0	...	49	324	
1095	ID_6c29eea22	160000.0	0	5	0	1	1	1	1.0	0	...	0	4	
1096	ID_870ae3993	160000.0	0	5	0	1	1	1	1.0	0	...	36	1444	
1097	ID_00da43675	160000.0	0	5	0	1	1	1	1.0	0	...	25	144	
1098	ID_22d124fdf	160000.0	0	5	0	1	1	1	1.0	0	...	64	361	
1099	ID_075c52143	160000.0	0	5	0	1	1	1	1.0	0	...	64	2025	

14 rows × 143 columns

In [131]:
```
# Is the dependency variable important?
trainDF.T.loc["dependency"].iloc[0:50]
```

Out[131]:
```
0      no
1       8
2       8
3     yes
4     yes
5     yes
6     yes
7     yes
8     yes
9     yes
10    yes
11    yes
12    yes
13    yes
14    yes
15      3
16      3
17      3
18      3
19     no
20     no
21      8
22      8
23     no
24     no
25     no
26     no
27     .5
28     .5
29     .5
30     no
31     no
32    .25
33    .25
34    .25
35    .25
36    .25
37     .5
38     .5
39     .5
40     no
41     no
42      2
43      2
44      2
45      8
46      8
47     .5
48     .5
49     .5
Name: dependency, dtype: object
```

In [132]:
```
# How many different values exists for the Target column?
trainDF["Target"].value_counts()
```

Out[132]:
```
4    5996
2    1597
3    1209
1     755
Name: Target, dtype: int64
```

In [133]:
```
trainDF.isna()
#trainDF.isna().sum()
```

Out[133]:

	Id	v2a1	hacdor	rooms	hacapo	v14a	refrig	v18q	v18q1	r4h1	...	SQBescolari	SQBage	SQBhogar_total
0	False	False	False	False	False	False	False	False	True	False	...	False	False	False
1	False	False	False	False	False	False	False	False	False	False	...	False	False	False
2	False	True	False	False	False	False	False	False	True	False	...	False	False	False
3	False	False	False	False	False	False	False	False	False	False	...	False	False	False
4	False	False	False	False	False	False	False	False	False	False	...	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9552	False	False	False	False	False	False	False	False	True	False	...	False	False	False
9553	False	False	False	False	False	False	False	False	True	False	...	False	False	False
9554	False	False	False	False	False	False	False	False	True	False	...	False	False	False
9555	False	False	False	False	False	False	False	False	True	False	...	False	False	False
9556	False	False	False	False	False	False	False	False	True	False	...	False	False	False

9557 rows × 143 columns

From the way things look, the output variable is "Target". It only appears in the training dataset, not the testing one. Target is a number between 1 and 4, which classifies the household.

In [134]:
```
# Determine if the columns "r4t3" and "tamhog" store the exact same data.

col1 = trainDF.loc[:, "r4t3"]
#print(col1)

col2 = trainDF.loc[:, "tamhog"]
#print(col2)

sumForAll = col1 - col2

type(sumForAll)
#print(sumForAll)

#sumForAll.sum()
```

Out[134]:
```
pandas.core.series.Series
```

In [135]:
```
# Examining how isolated columns affect the value in the Target column.

#trainDF.loc[:, ["r4t3", "tamhog", "Target"]].iloc[100: 150]
trainDF.loc[:, ["r4t3", "tamhog", "hhsize", "dis", "idhogar", "instlevel5", "hogar_nin", "hogar_adul", "hogar_mayor", "hogar_total", "dependency", "Target"]].iloc[0: 60]
```

Out[135]:

	r4t3	tamhog	hhsize	dis	idhogar	instlevel5	hogar_nin	hogar_adul	hogar_mayor	hogar_total	dependency	Targ
0	1	1	1	0	21eb7fcc1	0	0	1	0	1	no	
1	1	1	1	0	0e5d7a658	0	0	1	1	1	8	
2	1	1	1	1	2c7317ea8	1	0	1	1	1	8	
3	4	4	4	0	2b58d945f	0	2	2	0	4	yes	
4	4	4	4	0	2b58d945f	1	2	2	0	4	yes	
5	4	4	4	0	2b58d945f	1	2	2	0	4	yes	
6	4	4	4	0	2b58d945f	0	2	2	0	4	yes	
7	4	4	4	0	d6dae86b7	0	2	2	0	4	yes	
8	4	4	4	0	d6dae86b7	0	2	2	0	4	yes	
9	4	4	4	0	d6dae86b7	1	2	2	0	4	yes	
10	4	4	4	0	d6dae86b7	0	2	2	0	4	yes	
11	2	2	2	0	bb2094100	0	1	1	0	2	yes	
12	2	2	2	0	bb2094100	1	1	1	0	2	yes	
13	2	2	2	0	c51f9c774	0	0	2	1	2	yes	
14	2	2	2	0	c51f9c774	0	0	2	1	2	yes	
15	4	4	4	0	6893e65ca	1	2	2	1	4	3	
16	4	4	4	0	6893e65ca	0	2	2	1	4	3	
17	4	4	4	0	6893e65ca	1	2	2	1	4	3	
18	4	4	4	0	6893e65ca	0	2	2	1	4	3	
19	2	2	2	0	d29058053	0	0	2	0	2	no	
20	2	2	2	0	d29058053	0	0	2	0	2	no	
21	2	2	2	0	ec0e8edce	0	0	2	2	2	8	
22	2	2	2	0	ec0e8edce	0	0	2	2	2	8	
23	3	3	3	0	3e16fab89	0	0	3	0	3	no	
24	3	3	3	0	3e16fab89	0	0	3	0	3	no	
25	3	3	3	0	3e16fab89	0	0	3	0	3	no	
26	1	1	1	0	1e84a2ac8	0	0	1	0	1	no	
27	3	3	3	0	759df0194	0	0	3	1	3	.5	
28	3	3	3	0	759df0194	0	0	3	1	3	.5	
29	3	3	3	0	759df0194	0	0	3	1	3	.5	
30	2	2	2	0	f2fcf00fd	0	0	2	0	2	no	
31	2	2	2	0	f2fcf00fd	1	0	2	0	2	no	
32	5	5	5	0	cb6bb28dd	0	1	4	0	5	.25	
33	5	5	5	0	cb6bb28dd	1	1	4	0	5	.25	
34	5	5	5	0	cb6bb28dd	0	1	4	0	5	.25	
35	5	5	5	0	cb6bb28dd	0	1	4	0	5	.25	
36	5	5	5	0	cb6bb28dd	0	1	4	0	5	.25	
37	3	3	3	0	cbf24a06c	0	1	2	0	3	.5	
38	3	3	3	0	cbf24a06c	0	1	2	0	3	.5	
39	3	3	3	0	cbf24a06c	0	1	2	0	3	.5	
40	2	2	2	0	922a1f87a	0	0	2	0	2	no	
41	2	2	2	0	922a1f87a	0	0	2	0	2	no	
42	3	3	3	0	a57a1f2f4	0	1	2	1	3	2	
43	3	3	3	0	a57a1f2f4	0	1	2	1	3	2	

	r4t3	tamhog	hhsize	dis	idhogar	instlevel5	hogar_nin	hogar_adul	hogar_mayor	hogar_total	dependency	Targ
44	3	3	3	0	a57a1f2f4	0	1	2	1	3	2	
45	2	2	2	0	9b084b23d	0	0	2	2	2	8	
46	2	2	2	1	9b084b23d	0	0	2	2	2	8	
47	3	3	3	0	6ddd55b3e	0	1	2	0	3	.5	
48	3	3	3	0	6ddd55b3e	0	1	2	0	3	.5	
49	3	3	3	0	6ddd55b3e	1	1	2	0	3	.5	
50	4	4	4	0	652a7ffa0	0	2	2	0	4	yes	
51	4	4	4	0	652a7ffa0	1	2	2	0	4	yes	
52	4	4	4	0	652a7ffa0	0	2	2	0	4	yes	
53	4	4	4	0	652a7ffa0	1	2	2	0	4	yes	
54	4	4	4	0	b9d64d752	0	2	2	0	4	yes	
55	4	4	4	0	b9d64d752	0	2	2	0	4	yes	
56	4	4	4	0	b9d64d752	0	2	2	0	4	yes	
57	4	4	4	0	b9d64d752	1	2	2	0	4	yes	
58	1	1	1	0	09a26c158	0	0	1	1	1	8	
59	2	2	2	0	e2f9717d3	0	0	2	0	2	no	

Let's show the entire training dataset using the transposed dataframe over multiple cells.

In [136]:
```
trainDF.T[0:50]
```

Out[136]:

	0	1	2	3	4	5	6
Id	ID_279628684	ID_f29eb3ddd	ID_68de51c94	ID_d671db89c	ID_d56d6f5f5	ID_ec05b1a7b	ID_e9e0c1100	ID_3
v2a1	190000.0	135000.0	NaN	180000.0	180000.0	180000.0	180000.0	
hacdor	0	0	0	0	0	0	0	
rooms	3	4	8	5	5	5	5	
hacapo	0	0	0	0	0	0	0	
v14a	1	1	1	1	1	1	1	
refrig	1	1	1	1	1	1	1	
v18q	0	1	0	1	1	1	1	
v18q1	NaN	1.0	NaN	1.0	1.0	1.0	1.0	
r4h1	0	0	0	0	0	0	0	
r4h2	1	1	0	2	2	2	2	
r4h3	1	1	0	2	2	2	2	
r4m1	0	0	0	1	1	1	1	
r4m2	0	0	1	1	1	1	1	
r4m3	0	0	1	2	2	2	2	
r4t1	0	0	0	1	1	1	1	
r4t2	1	1	1	3	3	3	3	
r4t3	1	1	1	4	4	4	4	
tamhog	1	1	1	4	4	4	4	
tamviv	1	1	1	4	4	4	4	
escolari	10	12	11	9	11	11	2	
rez_esc	NaN	NaN	NaN	1.0	NaN	NaN	0.0	
hhsize	1	1	1	4	4	4	4	
paredblolad	1	0	0	1	1	1	1	
paredzocalo	0	0	0	0	0	0	0	
paredpreb	0	0	0	0	0	0	0	
pareddes	0	0	0	0	0	0	0	
paredmad	0	1	1	0	0	0	0	
paredzinc	0	0	0	0	0	0	0	
paredfibras	0	0	0	0	0	0	0	
paredother	0	0	0	0	0	0	0	
pisomoscer	1	0	1	1	1	1	1	
pisocemento	0	0	0	0	0	0	0	
pisoother	0	0	0	0	0	0	0	
pisonatur	0	0	0	0	0	0	0	
pisonotiene	0	0	0	0	0	0	0	
pisomadera	0	1	0	0	0	0	0	
techozinc	0	1	1	1	1	1	1	
techoentrepiso	1	0	0	0	0	0	0	
techocane	0	0	0	0	0	0	0	
techootro	0	0	0	0	0	0	0	
cielorazo	1	1	1	1	1	1	1	
abastaguadentro	1	1	1	1	1	1	1	
abastaguafuera	0	0	0	0	0	0	0	

	0	1	2	3	4	5	6
abastaguano	0	0	0	0	0	0	0
public	1	1	1	1	1	1	1
planpri	0	0	0	0	0	0	0
noelec	0	0	0	0	0	0	0
coopele	0	0	0	0	0	0	0
sanitario1	0	0	0	0	0	0	0

50 rows × 9557 columns

In [137]:
```
trainDF.T[50:100]
```

Out[137]:

	0	1	2	3	4	5	6	7	8
sanitario2	1	1	1	1	1	1	1	1	1
sanitario3	0	0	0	0	0	0	0	0	0
sanitario5	0	0	0	0	0	0	0	0	0
sanitario6	0	0	0	0	0	0	0	0	0
energcocinar1	0	0	0	0	0	0	0	0	0
energcocinar2	0	1	1	1	1	1	1	0	0
energcocinar3	1	0	0	0	0	0	0	1	1
energcocinar4	0	0	0	0	0	0	0	0	0
elimbasu1	1	1	1	1	1	1	1	1	1
elimbasu2	0	0	0	0	0	0	0	0	0
elimbasu3	0	0	0	0	0	0	0	0	0
elimbasu4	0	0	0	0	0	0	0	0	0
elimbasu5	0	0	0	0	0	0	0	0	0
elimbasu6	0	0	0	0	0	0	0	0	0
epared1	0	0	0	0	0	0	0	1	1
epared2	1	1	1	0	0	0	0	0	0
epared3	0	0	0	1	1	1	1	0	0
etecho1	1	0	0	0	0	0	0	1	1
etecho2	0	1	0	0	0	0	0	0	0
etecho3	0	0	1	1	1	1	1	0	0
eviv1	1	0	0	0	0	0	0	0	0
eviv2	0	1	0	0	0	0	0	1	1
eviv3	0	0	1	1	1	1	1	0	0
dis	0	0	1	0	0	0	0	0	0
male	1	1	0	1	0	1	0	0	1
female	0	0	1	0	1	0	1	1	0
estadocivil1	0	0	0	0	0	0	1	1	0
estadocivil2	0	0	0	0	1	1	0	0	1
estadocivil3	0	0	0	0	0	0	0	0	0
estadocivil4	1	1	0	0	0	0	0	0	0
estadocivil5	0	0	0	0	0	0	0	0	0
estadocivil6	0	0	1	0	0	0	0	0	0
estadocivil7	0	0	0	1	0	0	0	0	0
parentesco1	1	1	1	0	0	1	0	0	1
parentesco2	0	0	0	0	1	0	0	0	0
parentesco3	0	0	0	1	0	0	1	1	0
parentesco4	0	0	0	0	0	0	0	0	0
parentesco5	0	0	0	0	0	0	0	0	0
parentesco6	0	0	0	0	0	0	0	0	0
parentesco7	0	0	0	0	0	0	0	0	0
parentesco8	0	0	0	0	0	0	0	0	0
parentesco9	0	0	0	0	0	0	0	0	0
parentesco10	0	0	0	0	0	0	0	0	0
parentesco11	0	0	0	0	0	0	0	0	0

	0	1	2	3	4	5	6	7	8	
parentesco12	0	0	0	0	0	0	0	0	0	
idhogar	21eb7fcc1	0e5d7a658	2c7317ea8	2b58d945f	2b58d945f	2b58d945f	2b58d945f	d6dae86b7	d6dae86b7	d6dae
hogar_nin	0	0	0	2	2	2	2	2	2	
hogar_adul	1	1	1	2	2	2	2	2	2	
hogar_mayor	0	1	1	0	0	0	0	0	0	
hogar_total	1	1	1	4	4	4	4	4	4	

50 rows × 9557 columns

In [138]:
```
trainDF.T[100:144]
```

Out[138]:

	0	1	2	3	4	5	6	7	8	9	...	9547	95
dependency	no	8	8	yes	yes	yes	yes	yes	yes	yes	...	.66666669	.666666
edjefe	10	12	no	11	11	11	11	9	9	9	...	2	
edjefa	no	no	11	no	no	no	no	no	no	no	...	no	
meaneduc	10.0	12.0	11.0	11.0	11.0	11.0	11.0	10.0	10.0	10.0	...	10.0	1(
instlevel1	0	0	0	0	0	0	0	1	0	0	...	0	
instlevel2	0	0	0	0	0	0	1	0	0	0	...	0	
instlevel3	0	0	0	0	0	0	0	0	0	0	...	0	
instlevel4	1	0	0	1	0	0	0	0	1	0	...	0	
instlevel5	0	0	1	0	1	1	0	0	0	1	...	1	
instlevel6	0	0	0	0	0	0	0	0	0	0	...	0	
instlevel7	0	0	0	0	0	0	0	0	0	0	...	0	
instlevel8	0	1	0	0	0	0	0	0	0	0	...	0	
instlevel9	0	0	0	0	0	0	0	0	0	0	...	0	
bedrooms	1	1	2	3	3	3	3	1	1	1	...	3	
overcrowding	1.0	1.0	0.5	1.333333	1.333333	1.333333	1.333333	4.0	4.0	4.0	...	2.333333	2.3333
tipovivi1	0	0	1	0	0	0	0	0	0	0	...	0	
tipovivi2	0	0	0	0	0	0	0	0	0	0	...	1	
tipovivi3	1	1	0	1	1	1	1	1	1	1	...	0	
tipovivi4	0	0	0	0	0	0	0	0	0	0	...	0	
tipovivi5	0	0	0	0	0	0	0	0	0	0	...	0	
computer	0	0	0	0	0	0	0	0	0	0	...	0	
television	0	0	0	0	0	0	0	0	0	0	...	0	
mobilephone	1	1	0	1	1	1	1	1	1	1	...	1	
qmobilephone	1	1	0	3	3	3	3	1	1	1	...	3	
lugar1	1	1	1	1	1	1	1	1	1	1	...	0	
lugar2	0	0	0	0	0	0	0	0	0	0	...	0	
lugar3	0	0	0	0	0	0	0	0	0	0	...	0	
lugar4	0	0	0	0	0	0	0	0	0	0	...	0	
lugar5	0	0	0	0	0	0	0	0	0	0	...	0	
lugar6	0	0	0	0	0	0	0	0	0	0	...	1	
area1	1	1	1	1	1	1	1	1	1	1	...	0	
area2	0	0	0	0	0	0	0	0	0	0	...	1	
age	43	67	92	17	37	38	8	7	30	28	...	23	
SQBescolari	100	144	121	81	121	121	4	0	81	121	...	121	1
SQBage	1849	4489	8464	289	1369	1444	64	49	900	784	...	529	3
SQBhogar_total	1	1	1	16	16	16	16	16	16	16	...	25	
SQBedjefe	100	144	0	121	121	121	121	81	81	81	...	4	
SQBhogar_nin	0	0	0	4	4	4	4	4	4	4	...	4	
SQBovercrowding	1.0	1.0	0.25	1.777778	1.777778	1.777778	1.777778	16.0	16.0	16.0	...	5.444444	5.4444
SQBdependency	0.0	64.0	64.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	...	0.444444	0.4444
SQBmeaned	100.0	144.0	121.0	121.0	121.0	121.0	121.0	100.0	100.0	100.0	...	100.0	10(
agesq	1849	4489	8464	289	1369	1444	64	49	900	784	...	529	3
Target	4	4	4	4	4	4	4	4	4	4	...	4	

43 rows × 9557 columns

Let us examine how each set of columns affects the target variable. We will divide the columns into groups based on their subjects, including educaiton, household size, appliances, and home location. Rows 1086-1100 show lots of different values of the Target variable, so lets's use those rows.

Columns pertaining to home ownership status

In [139]:
```
# Columns pertaining to home ownership status. Columns 2, 116-120

#trainDF.loc[:, ["v2a1", "tipovivi1", "tipovivi2", "tipovivi3", "tipovivi4", "tipovivi5", "Targe
t"]].iloc[0: 60]
#trainDF.loc[:, ["v2a1", "tipovivi1", "tipovivi2", "tipovivi3", "tipovivi4", "tipovivi5", "Targe
t"]].iloc[60: 120]
#trainDF.loc[:, ["v2a1", "tipovivi1", "tipovivi2", "tipovivi3", "tipovivi4", "tipovivi5", "Targe
t"]].iloc[1050: 1100]
trainDF.loc[:, ["Id", "idhogar", "v2a1", "tipovivi1", "tipovivi2", "tipovivi3", "tipovivi4", "tipov
ivi5", "Target"]].iloc[1086: 1100]
```

Out[139]:

	Id	idhogar	v2a1	tipovivi1	tipovivi2	tipovivi3	tipovivi4	tipovivi5	Target
1086	ID_30d6500c3	2e65e4af3	150000.0	0	0	1	0	0	4
1087	ID_4203fd5e2	ee3d80cb6	NaN	1	0	0	0	0	2
1088	ID_7c33c1884	a0695cb68	NaN	1	0	0	0	0	1
1089	ID_77cce5b75	62c28e034	NaN	1	0	0	0	0	4
1090	ID_df05b9e45	62c28e034	NaN	1	0	0	0	0	4
1091	ID_3490fd704	8e284abd5	NaN	1	0	0	0	0	4
1092	ID_d82b1a426	8e284abd5	NaN	1	0	0	0	0	4
1093	ID_9d6957e8a	8e284abd5	NaN	1	0	0	0	0	4
1094	ID_5038a636a	304731467	160000.0	0	0	1	0	0	2
1095	ID_6c29eea22	304731467	160000.0	0	0	1	0	0	2
1096	ID_870ae3993	304731467	160000.0	0	0	1	0	0	2
1097	ID_00da43675	304731467	160000.0	0	0	1	0	0	2
1098	ID_22d124fdf	304731467	160000.0	0	0	1	0	0	2
1099	ID_075c52143	304731467	160000.0	0	0	1	0	0	2

In [140]:
```
# Are there any families with a precarious housing situation? These would have tipovivi4 = 1
trainDF["tipovivi4"].value_counts()
```

Out[140]:
```
0    9394
1     163
Name: tipovivi4, dtype: int64
```

In [141]:
```
# Check all columns with tipovivi4 = 1. Check if the target variable is 4, indicating the highest need possible.
trainDF[trainDF['tipovivi4'] == 1]
```

Out[141]:

	Id	v2a1	hacdor	rooms	hacapo	v14a	refrig	v18q	v18q1	r4h1	...	SQBescolari	SQBage	SQBhogar
245	ID_694d4b0ce	NaN	0	3	0	1	1	0	NaN	0	...	9	4225	
246	ID_db3aa162f	NaN	0	3	0	1	1	0	NaN	0	...	36	1600	
247	ID_92729a397	NaN	0	4	0	1	0	0	NaN	0	...	0	2025	
248	ID_d61734d32	NaN	0	4	0	1	0	0	NaN	0	...	36	3481	
249	ID_f1791bbeb	NaN	0	4	0	1	0	0	NaN	0	...	0	1764	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
9237	ID_c128541ee	NaN	0	4	0	1	1	0	NaN	2	...	4	64	
9238	ID_5b75bcf1c	NaN	0	4	0	1	1	0	NaN	2	...	49	196	
9239	ID_3bf2c6cff	NaN	0	4	0	1	1	0	NaN	2	...	36	961	
9240	ID_6d6a0b868	NaN	0	4	0	1	1	0	NaN	2	...	121	576	
9241	ID_bf737f17d	NaN	0	4	0	1	1	0	NaN	2	...	0	49	

163 rows × 143 columns

Columns pertaining to number of rooms and overcrowding

In [142]:
```
# Columns pertaining to number of rooms and overcrowding. Columns 3-6, 114, 115

#trainDF.loc[:, ["hacdor", "rooms", "hacapo", "bedrooms", "overcrowding", "Target"]].iloc[0: 60]
#trainDF.loc[:, ["hacdor", "rooms", "hacapo", "bedrooms", "overcrowding", "Target"]].iloc[60: 120]
#trainDF.loc[:, ["hacdor", "rooms", "hacapo", "bedrooms", "overcrowding", "Target"]].iloc[1050: 1100]
trainDF.loc[:, ["Id", "idhogar", "hacdor", "rooms", "hacapo", "bedrooms", "overcrowding", "Target"]].iloc[1086: 1100]
```

Out[142]:

	Id	idhogar	hacdor	rooms	hacapo	bedrooms	overcrowding	Target
1086	ID_30d6500c3	2e65e4af3	0	4	0	2	2.500000	4
1087	ID_4203fd5e2	ee3d80cb6	0	3	0	1	1.000000	2
1088	ID_7c33c1884	a0695cb68	0	6	0	3	0.333333	1
1089	ID_77cce5b75	62c28e034	0	4	0	2	1.000000	4
1090	ID_df05b9e45	62c28e034	0	4	0	2	1.000000	4
1091	ID_3490fd704	8e284abd5	0	7	0	4	2.500000	4
1092	ID_d82b1a426	8e284abd5	0	7	0	4	2.500000	4
1093	ID_9d6957e8a	8e284abd5	0	7	0	4	2.500000	4
1094	ID_5038a636a	304731467	0	5	0	3	2.000000	2
1095	ID_6c29eea22	304731467	0	5	0	3	2.000000	2
1096	ID_870ae3993	304731467	0	5	0	3	2.000000	2
1097	ID_00da43675	304731467	0	5	0	3	2.000000	2
1098	ID_22d124fdf	304731467	0	5	0	3	2.000000	2
1099	ID_075c52143	304731467	0	5	0	3	2.000000	2

Columns pertaining to number and type of people in home

In [143]:
```
# Columns pertaining to number and type of people in home. Columns 10-20, 23

#trainDF.loc[:, ["r4h1", "r4h2", "r4h3", "r4m1", "r4m2", "r4m3", "r4t1", "r4t2", "r4t3", "tamhog",
 "tamviv", "hhsize", "Target"]].iloc[0: 60]
#trainDF.loc[:, ["r4h1", "r4h2", "r4h3", "r4m1", "r4m2", "r4m3", "r4t1", "r4t2", "r4t3", "tamhog",
 "tamviv", "hhsize", "Target"]].iloc[60: 120]
#trainDF.loc[:, ["r4h1", "r4h2", "r4h3", "r4m1", "r4m2", "r4m3", "r4t1", "r4t2", "r4t3", "tamhog",
 "tamviv", "hhsize", "Target"]].iloc[1050: 1100]
trainDF.loc[:, ["Id", "idhogar", "r4h1", "r4h2", "r4h3", "r4m1", "r4m2", "r4m3", "r4t1", "r4t2", "r
4t3", "tamhog", "tamviv", "hhsize", "Target"]].iloc[1086: 1100]
```

Out[143]:

	Id	idhogar	r4h1	r4h2	r4h3	r4m1	r4m2	r4m3	r4t1	r4t2	r4t3	tamhog	tamviv	hhsize	Target
1086	ID_30d6500c3	2e65e4af3	1	2	3	0	2	2	1	4	5	5	5	5	4
1087	ID_4203fd5e2	ee3d80cb6	0	1	1	0	0	0	0	1	1	1	1	1	2
1088	ID_7c33c1884	a0695cb68	0	0	0	0	1	1	0	1	1	1	1	1	1
1089	ID_77cce5b75	62c28e034	0	1	1	0	1	1	0	2	2	2	2	2	4
1090	ID_df05b9e45	62c28e034	0	1	1	0	1	1	0	2	2	2	2	2	4
1091	ID_3490fd704	8e284abd5	0	1	1	0	2	2	0	3	3	3	10	3	4
1092	ID_d82b1a426	8e284abd5	0	1	1	0	2	2	0	3	3	3	10	3	4
1093	ID_9d6957e8a	8e284abd5	0	1	1	0	2	2	0	3	3	3	10	3	4
1094	ID_5038a636a	304731467	0	3	3	1	2	3	1	5	6	6	6	6	2
1095	ID_6c29eea22	304731467	0	3	3	1	2	3	1	5	6	6	6	6	2
1096	ID_870ae3993	304731467	0	3	3	1	2	3	1	5	6	6	6	6	2
1097	ID_00da43675	304731467	0	3	3	1	2	3	1	5	6	6	6	6	2
1098	ID_22d124fdf	304731467	0	3	3	1	2	3	1	5	6	6	6	6	2
1099	ID_075c52143	304731467	0	3	3	1	2	3	1	5	6	6	6	6	2

In [144]:
```
# Columns pertaining to number and type of people in home. Columns 18, 84-95.

#trainDF.loc[:, ["parentesco1", "parentesco2", "parentesco3", "parentesco4", "parentesco4", "parentesco5", "parentesco6", "parentesco7", "parentesco8", "parentesco9", "parentesco10", "parentesco11", "parentesco12", "Target"]].iloc[0: 60]
#trainDF.loc[:, ["parentesco1", "parentesco2", "parentesco3", "parentesco4", "parentesco4", "parentesco5", "parentesco6", "parentesco7", "parentesco8", "parentesco9", "parentesco10", "parentesco11", "parentesco12", "Target"]].iloc[60: 120]
#trainDF.loc[:, ["parentesco1", "parentesco2", "parentesco3", "parentesco4", "parentesco4", "parentesco5", "parentesco6", "parentesco7", "parentesco8", "parentesco9", "parentesco10", "parentesco11", "parentesco12", "Target"]].iloc[1050: 1100]
trainDF.loc[:, ["Id", "idhogar", "r4t3", "parentesco1", "parentesco2", "parentesco3", "parentesco4", "parentesco4", "parentesco5", "parentesco6", "parentesco7", "parentesco8", "parentesco9", "parentesco10", "parentesco11", "parentesco12", "Target"]].iloc[1086: 1100]
```

Out[144]:

	Id	idhogar	r4t3	parentesco1	parentesco2	parentesco3	parentesco4	parentesco4	parentesco5	pare
1086	ID_30d6500c3	2e65e4af3	5	0	0	0	0	0	0	
1087	ID_4203fd5e2	ee3d80cb6	1	1	0	0	0	0	0	
1088	ID_7c33c1884	a0695cb68	1	1	0	0	0	0	0	
1089	ID_77cce5b75	62c28e034	2	0	0	0	0	0	0	
1090	ID_df05b9e45	62c28e034	2	1	0	0	0	0	0	
1091	ID_3490fd704	8e284abd5	3	0	0	0	0	0	1	
1092	ID_d82b1a426	8e284abd5	3	1	0	0	0	0	0	
1093	ID_9d6957e8a	8e284abd5	3	0	0	1	0	0	0	
1094	ID_5038a636a	304731467	6	0	0	1	0	0	0	
1095	ID_6c29eea22	304731467	6	0	0	1	0	0	0	
1096	ID_870ae3993	304731467	6	0	1	0	0	0	0	
1097	ID_00da43675	304731467	6	0	0	1	0	0	0	
1098	ID_22d124fdf	304731467	6	0	0	1	0	0	0	
1099	ID_075c52143	304731467	6	1	0	0	0	0	0	

In [145]:
```
# Columns pertaining to number and type of people in home. Columns 74-76, 97-100, 101, 133

#trainDF.loc[:, ["dis", "male", "female", "hogar_nin", "hogar_adul", "hogar_mayor", "hogar_total",
 "dependency", "age", "Target"]].iloc[0: 60]
#trainDF.loc[:, ["dis", "male", "female", "hogar_nin", "hogar_adul", "hogar_mayor", "hogar_total",
 "dependency", "age", "Target"]].iloc[60: 120]
#trainDF.loc[:, ["Id", "idhogar", "dis", "male", "female", "hogar_nin", "hogar_adul", "hogar_mayo
r", "hogar_total", "dependency", "age", "Target"]].iloc[1050: 1100]
trainDF.loc[:, ["Id", "idhogar", "dis", "male", "female", "hogar_nin", "hogar_adul", "hogar_mayor",
"hogar_total", "dependency", "age", "Target"]].iloc[1086: 1100]
```

Out[145]:

	Id	idhogar	dis	male	female	hogar_nin	hogar_adul	hogar_mayor	hogar_total	dependency	age	T
1086	ID_30d6500c3	2e65e4af3	0	0	1	1	4	0	5	.25	63	
1087	ID_4203fd5e2	ee3d80cb6	0	1	0	0	1	1	1	8	77	
1088	ID_7c33c1884	a0695cb68	0	0	1	0	1	0	1	no	60	
1089	ID_77cce5b75	62c28e034	1	1	0	0	2	2	2	8	66	
1090	ID_df05b9e45	62c28e034	0	0	1	0	2	2	2	8	67	
1091	ID_3490fd704	8e284abd5	0	0	1	0	3	0	3	no	20	
1092	ID_d82b1a426	8e284abd5	0	0	1	0	3	0	3	no	52	
1093	ID_9d6957e8a	8e284abd5	0	1	0	0	3	0	3	no	22	
1094	ID_5038a636a	304731467	0	1	0	4	2	0	6	2	18	
1095	ID_6c29eea22	304731467	0	0	1	4	2	0	6	2	2	
1096	ID_870ae3993	304731467	0	0	1	4	2	0	6	2	38	
1097	ID_00da43675	304731467	0	0	1	4	2	0	6	2	12	
1098	ID_22d124fdf	304731467	0	1	0	4	2	0	6	2	19	
1099	ID_075c52143	304731467	0	1	0	4	2	0	6	2	45	

Columns pertaining to house material quality

In [146]:
```
# Columns pertaining to house material quality. Columns 24-31, 32-35, 37, 36

#trainDF.loc[:, ["paredblolad", "paredzocalo", "paredpreb", "pareddes", "paredmad", "paredzinc", "p
aredfibras", "paredother", "pisomoscer", "pisocemento", "pisoother", "pisonatur", "pisomadera", "pi
sonotiene", "Target"]].iloc[0: 60]
#trainDF.loc[:, ["paredblolad", "paredzocalo", "paredpreb", "pareddes", "paredmad", "paredzinc", "p
aredfibras", "paredother", "pisomoscer", "pisocemento", "pisoother", "pisonatur", "pisomadera", "pi
sonotiene", "Target"]].iloc[1050: 1100]
trainDF.loc[:, ["Id", "idhogar", "paredblolad", "paredzocalo", "paredpreb", "pareddes", "paredmad",
"paredzinc", "paredfibras", "paredother", "pisomoscer", "pisocemento", "pisoother", "pisonatur", "p
isomadera", "pisonotiene", "Target"]].iloc[1086: 1100]
```

Out[146]:

	Id	idhogar	paredblolad	paredzocalo	paredpreb	pareddes	paredmad	paredzinc	paredfibras	paredo
1086	ID_30d6500c3	2e65e4af3	1	0	0	0	0	0	0	
1087	ID_4203fd5e2	ee3d80cb6	0	1	0	0	0	0	0	
1088	ID_7c33c1884	a0695cb68	0	0	0	0	1	0	0	
1089	ID_77cce5b75	62c28e034	1	0	0	0	0	0	0	
1090	ID_df05b9e45	62c28e034	1	0	0	0	0	0	0	
1091	ID_3490fd704	8e284abd5	1	0	0	0	0	0	0	
1092	ID_d82b1a426	8e284abd5	1	0	0	0	0	0	0	
1093	ID_9d6957e8a	8e284abd5	1	0	0	0	0	0	0	
1094	ID_5038a636a	304731467	1	0	0	0	0	0	0	
1095	ID_6c29eea22	304731467	1	0	0	0	0	0	0	
1096	ID_870ae3993	304731467	1	0	0	0	0	0	0	
1097	ID_00da43675	304731467	1	0	0	0	0	0	0	
1098	ID_22d124fdf	304731467	1	0	0	0	0	0	0	
1099	ID_075c52143	304731467	1	0	0	0	0	0	0	

In [147]:
```
# Columns pertaining to house material quality. Columns 38-41, 42, 65-67, 68-70, 71-73

#trainDF.loc[:, ["techozinc", "techoentrepiso", "techocane", "techootro", "cielorazo", "epared1",
 "epared2", "epared3", "etecho1", "etecho2", "etecho3", "eviv1", "eviv2", "eviv3", "Target"]].iloc
[0: 60]
#trainDF.loc[:, ["techozinc", "techoentrepiso", "techocane", "techootro", "cielorazo", "epared1",
 "epared2", "epared3", "etecho1", "etecho2", "etecho3", "eviv1", "eviv2", "eviv3", "Target"]].iloc
[1050: 1100]
trainDF.loc[:, ["Id", "idhogar", "techozinc", "techoentrepiso", "techocane", "techootro", "cielorazo", "epared1", "epared2", "epared3", "etecho1", "etecho2", "etecho3", "eviv1", "eviv2", "eviv3", "Target"]].iloc[1086: 1100]
```

Out[147]:

	Id	idhogar	techozinc	techoentrepiso	techocane	techootro	cielorazo	epared1	epared2	epared3	et
1086	ID_30d6500c3	2e65e4af3	1	0	0	0	1	0	0	1	
1087	ID_4203fd5e2	ee3d80cb6	1	0	0	0	1	0	0	1	
1088	ID_7c33c1884	a0695cb68	1	0	0	0	1	0	0	1	
1089	ID_77cce5b75	62c28e034	1	0	0	0	1	0	0	1	
1090	ID_df05b9e45	62c28e034	1	0	0	0	1	0	0	1	
1091	ID_3490fd704	8e284abd5	1	0	0	0	0	0	1	0	
1092	ID_d82b1a426	8e284abd5	1	0	0	0	0	0	1	0	
1093	ID_9d6957e8a	8e284abd5	1	0	0	0	0	0	1	0	
1094	ID_5038a636a	304731467	1	0	0	0	1	0	1	0	
1095	ID_6c29eea22	304731467	1	0	0	0	1	0	1	0	
1096	ID_870ae3993	304731467	1	0	0	0	1	0	1	0	
1097	ID_00da43675	304731467	1	0	0	0	1	0	1	0	
1098	ID_22d124fdf	304731467	1	0	0	0	1	0	1	0	
1099	ID_075c52143	304731467	1	0	0	0	1	0	1	0	

Columns pertaining to necessary facilities and appliances for good life quality

In [148]:
```
# Columns pertaining to necessary facilities and appliances for good life quality. 7, 43-45, 46-49, 50-54

trainDF.loc[:, ["Id", "idhogar", "refrig", "abastaguadentro", "abastaguafuera", "abastaguano", "public", "planpri", "noelec", "coopele", "sanitario1", "sanitario2", "sanitario3", "sanitario5", "sanitario6", "Target"]].iloc[1086: 1100]
```

Out[148]:

	Id	idhogar	refrig	abastaguadentro	abastaguafuera	abastaguano	public	planpri	noelec	coopele	s
1086	ID_30d6500c3	2e65e4af3	1	1	0	0	1	0	0	0	
1087	ID_4203fd5e2	ee3d80cb6	1	1	0	0	1	0	0	0	
1088	ID_7c33c1884	a0695cb68	1	1	0	0	1	0	0	0	
1089	ID_77cce5b75	62c28e034	1	1	0	0	1	0	0	0	
1090	ID_df05b9e45	62c28e034	1	1	0	0	1	0	0	0	
1091	ID_3490fd704	8e284abd5	1	1	0	0	1	0	0	0	
1092	ID_d82b1a426	8e284abd5	1	1	0	0	1	0	0	0	
1093	ID_9d6957e8a	8e284abd5	1	1	0	0	1	0	0	0	
1094	ID_5038a636a	304731467	1	1	0	0	1	0	0	0	
1095	ID_6c29eea22	304731467	1	1	0	0	1	0	0	0	
1096	ID_870ae3993	304731467	1	1	0	0	1	0	0	0	
1097	ID_00da43675	304731467	1	1	0	0	1	0	0	0	
1098	ID_22d124fdf	304731467	1	1	0	0	1	0	0	0	
1099	ID_075c52143	304731467	1	1	0	0	1	0	0	0	

In [149]:
```
# Columns pertaining to necessary facilities and appliances for good life quality. 55-58, 59-64

trainDF.loc[:, ["Id", "idhogar", "energcocinar1", "energcocinar2", "energcocinar3", "energcocinar4", "elimbasu1", "elimbasu2", "elimbasu3", "elimbasu4", "elimbasu5", "elimbasu6", "Target"]].iloc[1086: 1100]
```

Out[149]:

	Id	idhogar	energcocinar1	energcocinar2	energcocinar3	energcocinar4	elimbasu1	elimbasu2	elimba
1086	ID_30d6500c3	2e65e4af3	0	1	0	0	1	0	
1087	ID_4203fd5e2	ee3d80cb6	0	1	0	0	1	0	
1088	ID_7c33c1884	a0695cb68	0	0	1	0	1	0	
1089	ID_77cce5b75	62c28e034	0	1	0	0	1	0	
1090	ID_df05b9e45	62c28e034	0	1	0	0	1	0	
1091	ID_3490fd704	8e284abd5	0	1	0	0	1	0	
1092	ID_d82b1a426	8e284abd5	0	1	0	0	1	0	
1093	ID_9d6957e8a	8e284abd5	0	1	0	0	1	0	
1094	ID_5038a636a	304731467	0	1	0	0	1	0	
1095	ID_6c29eea22	304731467	0	1	0	0	1	0	
1096	ID_870ae3993	304731467	0	1	0	0	1	0	
1097	ID_00da43675	304731467	0	1	0	0	1	0	
1098	ID_22d124fdf	304731467	0	1	0	0	1	0	
1099	ID_075c52143	304731467	0	1	0	0	1	0	

Columns pertaining to education

In [150]:
```
# Columns pertaining to education. Columns 21-22, 105-113

trainDF.loc[:, ["Id", "idhogar", "escolari", "rez_esc", "instlevel1", "instlevel2", "instlevel3",
"instlevel4", "instlevel5", "instlevel6", "instlevel7", "instlevel8", "instlevel9", "Target"]].iloc
[1086: 1100]
```

Out[150]:

	Id	idhogar	escolari	rez_esc	instlevel1	instlevel2	instlevel3	instlevel4	instlevel5	instlevel6	instlev
1086	ID_30d6500c3	2e65e4af3	9	NaN	0	0	0	1	0	0	
1087	ID_4203fd5e2	ee3d80cb6	7	NaN	0	0	0	1	0	0	
1088	ID_7c33c1884	a0695cb68	16	NaN	0	0	0	0	0	0	
1089	ID_77cce5b75	62c28e034	3	NaN	0	1	0	0	0	0	
1090	ID_df05b9e45	62c28e034	2	NaN	0	1	0	0	0	0	
1091	ID_3490fd704	8e284abd5	11	NaN	0	0	0	0	0	1	
1092	ID_d82b1a426	8e284abd5	9	NaN	0	0	0	0	0	1	
1093	ID_9d6957e8a	8e284abd5	11	NaN	0	0	0	0	1	0	
1094	ID_5038a636a	304731467	7	NaN	0	0	0	1	0	0	
1095	ID_6c29eea22	304731467	0	NaN	1	0	0	0	0	0	
1096	ID_870ae3993	304731467	6	NaN	0	0	1	0	0	0	
1097	ID_00da43675	304731467	5	0.0	0	1	0	0	0	0	
1098	ID_22d124fdf	304731467	8	NaN	0	0	0	1	0	0	
1099	ID_075c52143	304731467	8	NaN	0	0	0	1	0	0	

Columns pertaining to household heads

In [151]:
```
# Columns pertaining to household heads. Columns 102-104, 77-83

trainDF.loc[:, ["Id", "idhogar", "edjefe", "edjefa", "meaneduc", "estadocivil1", "estadocivil2", "e
stadocivil3", "estadocivil4", "estadocivil5", "estadocivil6", "estadocivil7", "Target"]].iloc[1086:
1100]
```

Out[151]:

	Id	idhogar	edjefe	edjefa	meaneduc	estadocivil1	estadocivil2	estadocivil3	estadocivil4	estadocivil5
1086	ID_30d6500c3	2e65e4af3	11	no	9.250000	0	0	0	1	0
1087	ID_4203fd5e2	ee3d80cb6	7	no	7.000000	0	0	0	0	1
1088	ID_7c33c1884	a0695cb68	no	16	16.000000	0	0	0	0	0
1089	ID_77cce5b75	62c28e034	no	2	2.500000	0	0	0	0	1
1090	ID_df05b9e45	62c28e034	no	2	2.500000	0	0	0	1	0
1091	ID_3490fd704	8e284abd5	no	9	10.333333	0	1	0	0	0
1092	ID_d82b1a426	8e284abd5	no	9	10.333333	0	0	0	1	0
1093	ID_9d6957e8a	8e284abd5	no	9	10.333333	0	1	0	0	0
1094	ID_5038a636a	304731467	8	no	14.500000	0	0	0	0	0
1095	ID_6c29eea22	304731467	8	no	14.500000	1	0	0	0	0
1096	ID_870ae3993	304731467	8	no	14.500000	0	0	1	0	0
1097	ID_00da43675	304731467	8	no	14.500000	0	0	0	0	0
1098	ID_22d124fdf	304731467	8	no	14.500000	0	0	0	0	0
1099	ID_075c52143	304731467	8	no	14.500000	0	0	1	0	0

Columns pertaining to home location

In [152]:
```
# Columns pertaining to household heads. Columns 125-130, 131-132

trainDF.loc[:, ["Id", "idhogar", "lugar1", "lugar2", "lugar3", "lugar4", "lugar5", "lugar6", "area
1", "area2", "Target"]].iloc[1086: 1100]
```

Out[152]:

	Id	idhogar	lugar1	lugar2	lugar3	lugar4	lugar5	lugar6	area1	area2	Target
1086	ID_30d6500c3	2e65e4af3	1	0	0	0	0	0	1	0	4
1087	ID_4203fd5e2	ee3d80cb6	1	0	0	0	0	0	1	0	2
1088	ID_7c33c1884	a0695cb68	1	0	0	0	0	0	1	0	1
1089	ID_77cce5b75	62c28e034	1	0	0	0	0	0	1	0	4
1090	ID_df05b9e45	62c28e034	1	0	0	0	0	0	1	0	4
1091	ID_3490fd704	8e284abd5	1	0	0	0	0	0	1	0	4
1092	ID_d82b1a426	8e284abd5	1	0	0	0	0	0	1	0	4
1093	ID_9d6957e8a	8e284abd5	1	0	0	0	0	0	1	0	4
1094	ID_5038a636a	304731467	1	0	0	0	0	0	1	0	2
1095	ID_6c29eea22	304731467	1	0	0	0	0	0	1	0	2
1096	ID_870ae3993	304731467	1	0	0	0	0	0	1	0	2
1097	ID_00da43675	304731467	1	0	0	0	0	0	1	0	2
1098	ID_22d124fdf	304731467	1	0	0	0	0	0	1	0	2
1099	ID_075c52143	304731467	1	0	0	0	0	0	1	0	2

Columns pertaining to electronics

In [153]:
```
# Columns pertaining to household heads. Columns 121-124, 8-9

trainDF.loc[:, ["Id", "idhogar", "computer", "television", "mobilephone", "qmobilephone", "v18q",
"v18q1", "Target"]].iloc[1086: 1100]
```

Out[153]:

	Id	idhogar	computer	television	mobilephone	qmobilephone	v18q	v18q1	Target
1086	ID_30d6500c3	2e65e4af3	0	0	1	7	1	1.0	4
1087	ID_4203fd5e2	ee3d80cb6	0	1	1	1	0	NaN	2
1088	ID_7c33c1884	a0695cb68	0	0	1	1	0	NaN	1
1089	ID_77cce5b75	62c28e034	0	0	0	0	0	NaN	4
1090	ID_df05b9e45	62c28e034	0	0	0	0	0	NaN	4
1091	ID_3490fd704	8e284abd5	0	1	1	6	0	NaN	4
1092	ID_d82b1a426	8e284abd5	0	1	1	6	0	NaN	4
1093	ID_9d6957e8a	8e284abd5	0	1	1	6	0	NaN	4
1094	ID_5038a636a	304731467	0	0	1	4	1	1.0	2
1095	ID_6c29eea22	304731467	0	0	1	4	1	1.0	2
1096	ID_870ae3993	304731467	0	0	1	4	1	1.0	2
1097	ID_00da43675	304731467	0	0	1	4	1	1.0	2
1098	ID_22d124fdf	304731467	0	0	1	4	1	1.0	2
1099	ID_075c52143	304731467	0	0	1	4	1	1.0	2

Check for biases in the data

In [154]:
```
# # Use this cell to check which columns in our datframe have the same value. We can ignore these c
olumns when checking for biases in the dataset.

# # Rows that we care about: 1086-1100.

# importantRows = np.arange(1086, 1101, 1)
# print(importantRows)

# # Loop over each column
# for column in trainDF.columns:
#     if(trainDF.select_dtypes(column))
#         print(column)

# # for index in importantRows:
# #     print(trainDF['Id'][index], " , " ,  trainDF['computer'][index])
```

In [155]:
```
# Check the purpose of the Id and idhogar columns
trainDF.loc[:, ["Id", "idhogar", "Target"]].iloc[1050: 1100]
```

Out[155]:

	Id	idhogar	Target
1050	ID_f5fb05e4b	bee362328	4
1051	ID_92ee60ec1	bee362328	4
1052	ID_93a1109cd	bee362328	4
1053	ID_814c4c286	bee362328	4
1054	ID_456c35118	35e03f915	1
1055	ID_23b14b70b	35e03f915	1
1056	ID_d0ae7df26	6712a6d7a	4
1057	ID_25c17b845	6712a6d7a	4
1058	ID_2b5ab3f13	6712a6d7a	4
1059	ID_8a2d63b83	eacd63a5b	4
1060	ID_ffb25c97c	807ef6114	3
1061	ID_7fb3908b0	807ef6114	3
1062	ID_431d4a3c8	cfcac85ee	4
1063	ID_92e32d527	cfcac85ee	4
1064	ID_32ab25463	cfcac85ee	4
1065	ID_2e03e62a8	cfcac85ee	4
1066	ID_8b4000559	cfcac85ee	4
1067	ID_5d3fb798b	504b957f9	4
1068	ID_e04a6c08b	504b957f9	4
1069	ID_ef0cdb14e	504b957f9	4
1070	ID_75f7cca97	ca2b17170	1
1071	ID_5a4e9b989	ca2b17170	1
1072	ID_2577568e4	ca2b17170	1
1073	ID_6d43e3af4	4f22c4256	4
1074	ID_717bcdcfb	4f22c4256	4
1075	ID_fa4c17ee4	2c420449d	4
1076	ID_2aaa738ce	5b0be63cb	4
1077	ID_ffdf8bdda	5b0be63cb	4
1078	ID_eba812b88	5bc8a6e4f	3
1079	ID_38c740c65	5bc8a6e4f	3
1080	ID_de26a6642	5bc8a6e4f	3
1081	ID_9e5bbae79	5bc8a6e4f	3
1082	ID_4174d5beb	2e65e4af3	4
1083	ID_6aae3924e	2e65e4af3	4
1084	ID_3bb173ba9	2e65e4af3	4
1085	ID_f6e6000fc	2e65e4af3	4
1086	ID_30d6500c3	2e65e4af3	4
1087	ID_4203fd5e2	ee3d80cb6	2
1088	ID_7c33c1884	a0695cb68	1
1089	ID_77cce5b75	62c28e034	4
1090	ID_df05b9e45	62c28e034	4
1091	ID_3490fd704	8e284abd5	4
1092	ID_d82b1a426	8e284abd5	4
1093	ID_9d6957e8a	8e284abd5	4

	Id	idhogar	Target
1094	ID_5038a636a	304731467	2
1095	ID_6c29eea22	304731467	2
1096	ID_870ae3993	304731467	2
1097	ID_00da43675	304731467	2
1098	ID_22d124fdf	304731467	2
1099	ID_075c52143	304731467	2

In [156]:
```
# Does the idhogar column keep track of each household as I think it does?

specificDF = trainDF[trainDF["idhogar"] == "5bc8a6e4f"]
specificDF[["Id", "hogar_total", "parentesco1", "parentesco2", "parentesco3", "parentesco4", "paren
tesco4", "parentesco5", "parentesco6", "parentesco7", "parentesco8", "parentesco9", "parentesco10",
"parentesco11", "parentesco12", "Target"]]
```

Out[156]:

	Id	hogar_total	parentesco1	parentesco2	parentesco3	parentesco4	parentesco4	parentesco5	parentesc
1078	ID_eba812b88	4	1	0	0	0	0	0	
1079	ID_38c740c65	4	0	0	1	0	0	0	
1080	ID_de26a6642	4	0	0	1	0	0	0	
1081	ID_9e5bbae79	4	0	1	0	0	0	0	

In [157]:
```
# How many households are there in total?

#trainDF["idhogar"].value_counts()
len(trainDF["idhogar"].unique())
```

Out[157]: 2988

Check if each family member in each household has the same poverty level.

In [158]:
```
# # Iterate through the entire dataframe row by row. Check each group of homes and see if the targe
t variable is the same for all family members.
# # Make a boolean list that is True for each household with consistent target values and False for
inconsistent values.

# # Get the unique household identifiers
# homeList = trainDF["idhogar"].unique()

# # Make a dictionary that contains the values of each home
# # Each key will be the idhogar string of the home. The value will be a list in the form [Boolean,
target1, target 2, ...]
# familyTargets =  dict.fromkeys(homeList, [True])

# #print(trainDF["idhogar"] == "21eb7fcc1")
# #print(homeList)
# #trainDF["idhogar"] == "304731467"

# # Updating key,value pair example
# #print(familyTargets)
# # key = "0e5d7a658"
# # value = familyTargets.get(key)
# # value.append(4)
# # familyTargets.update({key: value})
# # print(familyTargets)

# # # Iterate by row
# # for index in trainDF.index:
# #     #print(trainDF['Id'][index], " , " ,  trainDF['idhogar'][index], " , " , trainDF["Target"]
[index])

# #     print(str(index) + " is the current index")

# #     # Get the current key
# #     currentKey = trainDF["idhogar"][index]
# #     print(currentKey + " is the key")

# #     # Get the Target value in the row
# #     currentTarget = trainDF["Target"][index]
# #     #print(currentTarget)

# #     # If the value of the dict at for the current key is a list of length 1, then we keep the b
oolean as true and append the target to the end of the list
# #     # If the value of the dict at for the current key is a list of length 2, check if the last
 value is the same as currentTarget. If so, change nothing. If not,
# #     # then append the currentTarget to the end of the end of the list and change the boolean to
False.
# #     # If the length of the list is 3 or more, the boolean is false and we check if the currentT
arget is in the list. If so, do nothing. If not, then append.

# #     # Get the dictionary list and its length.
# #     valueList = familyTargets.get(currentKey)
# #     print(str(valueList) + " is current value list")
# #     lenList = len(valueList)

# #     # Care for length 1 lists. No taget values appended yet
# #     if(lenList == 1):
# #         print("Entered length 1 conditional")
# #         valueList.append(currentTarget)
# #         familyTargets.update({currentKey: valueList}) # Make the new list the current value
# #         #print(familyTargets[currentKey])
# #         #print(familyTargets["304731467"])
# #         print("Appended to length 1 list")

# #     print()

# #     #currentKey = familyTargets[trainDF["idhogar"][index]]
```

```
# # for home in homeList:
# #     if(trainDF["idhogar"] == home):
# #         print(trainDF["Target"])
```

In [159]:
```
# Attempt 2 of determining consistent poverty levels

# If we islate each household, take the average and mode of the target variable, and divide the for
mer by the latter, the result should 1.0.
# If the result is NOT 1.0, then the poverty levels in that particular household are inconsistent a
mong family members.

# List of all home IDs
homeList = trainDF["idhogar"].unique()

# Variable to store the total normalized average of all the homes.
sumofNormAverages = 0

# List to contain home IDs for homes with differing poverty levels
inconsistentPovertyHomes = []

# Iterate over homes
for home in homeList:
    #print("New iteration")

    # Get an isolated DF for just that home
    isolatedDF = trainDF[trainDF["idhogar"] == home]

    # Get the average and mode of the Target column for this isolated DF
    average = isolatedDF["Target"].mean()
    #print(average)
    mode = isolatedDF["Target"].mode()
    #print(mode)

    # The normalized average should 1.0 in all cases where family members have the same poverty lev
el.
    normalizedAverage = (average/mode)[0]

    #print(normalizedAverage)

    # Add the abnormal home IDs to the list we made before
    if(normalizedAverage != 1.0):
        #print(home)
        inconsistentPovertyHomes.append(home)

    # Add the normalized average to the list of all normalized averages
    sumofNormAverages += normalizedAverage

    #print()


# If all familes have the same poverty level, this value should be 1.0
print(sumofNormAverages / len(homeList))
```

```
1.0009487746561039
```

In [160]:
```
# Output the list of homes with inconsistent poverty levels. Output its length, too
print(len(inconsistentPovertyHomes))
print(inconsistentPovertyHomes)
```

```
85
['4b6077882', '6833ac5dc', '43b9c83e5', '5c3f7725d', '0f9494d3a', 'daafc1281', '73d85d05d', 'bcaa2
e2f5', '44f219a16', 'efd3aec61', '3c6973219', '0511912b6', 'f006348ed', 'a20ff33ba', '5e9329fc6',
'e65d4b943', '42ec8bef5', '6bcf799cf', '26b3a0f41', '4dc11e11f', '594d3eb27', 'd9b1558b5', '7ea6ac
a15', '8bb6da3c1', '3df651058', '811a35744', '2cb443214', 'bcab69521', '694a0cbf4', '3fe29a56b',
'636330516', '288579c97', '15a891635', '6a389f3de', 'a3288e6fa', '4e19bd549', '80a66379b', '5c6f32
bbc', '932287f5d', 'bd82509d1', '614b48fb7', '46af47063', '6c543442a', '410194c8b', '417865404',
'f7b421c2c', '67ad49822', '17fb04a62', 'c38913488', '513adb616', 'dfb966eec', '30a70901d', '18832b
840', '7c57f8237', 'c13325faf', '54118d5d9', '0f3e65c83', '03f4e5f4d', '8ae3e74ca', '309fb7246',
'09e25d616', '564eab113', '8242a51ec', '0172ab1d9', 'a94a45642', 'be91da044', '50e064ee8', '4c2dba
109', '7ad269eef', '3c73c107f', '55a662731', 'e17b252ed', '078a0b6e2', '28893b5e7', 'd64524b6b',
'2c9872b82', 'f94589d38', '8420bcfca', '71cd52a80', '654ef7612', 'cc971b690', '7e9d58c5c', 'e235a4
eec', 'c7ce4e30c', '9bbf7c6ca']
```

Check if there is a house without a family head. We can use similar code from the previous cell

In [161]:
```
# The family head should have parentesco1 = 1. Every idhogar value should have one row where parent
esco1 = 1. Those that do not indicate homes that do not have family heads.

# List of all households.
homeList = trainDF["idhogar"].unique()

# boolean to keep track of prescence of family head
familyHeadExists = False

# List to hold all homes without family heads
familiesWOutHeads = []

# Iterate over all households
for home in homeList:
    #print("New iteration")

    # Get an isolated DF for just that home
    isolatedDF = trainDF[trainDF["idhogar"] == home]

    # Iteratve over the isolate cell and check the value of parentesco1 in each row. If parentesco1
= 1, then set our boolean tracker to True.
    for index in isolatedDF.index:
        if(trainDF["parentesco1"][index] == 1):
            familyHeadExists = True

    # If the boolean is False, add the home ID to the familiesWOutHeads list
    if(familyHeadExists == False):
        familiesWOutHeads.append(home)

    # Set the boolean to False again for the next loop
    familyHeadExists = False
```

In [162]:
```
# Print the list of families without heads
print(len(familiesWOutHeads))
print(familiesWOutHeads)
```

```
15
['09b195e7a', '896fe6d3e', '61c10e099', '374ca5a19', 'bfd5067c2', '1367ab31d', '6b1b2405f', 'f2bfa
75c4', '03c6bdf85', 'ad687ad89', 'b1f4d89d7', 'c0c8a5013', 'a0812ef17', 'd363d9183', '1bc617b23']
```

Count how many null values are existing in columns.

In [163]:
```
# Iterate over every column and use isnull().sum() to get the total number if null values.

# Make a series null values for each column
nullSeries = trainDF.isnull().sum()

# Add all values of the null series together to get the total number of null values

total = 0

for item in nullSeries:
    total += item

print(total)
```

```
22140
```

Remove null value rows of the target variable.

In [164]:
```
# Are there any null values in the target column?
trainDF["Target"].isnull().any()
```

Out[164]:
```
False
```

In [165]:
```
# Are there any null values in the target column?
trainDF["Target"].isna().any()
```

Out[165]:
```
False
```

Predict the accuracy using random forest classifier

In [166]:
```
# Import train test split and Random forest model
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
```

In [167]:
```
# Perform train test split. The x-values should be everything not in the target column. The y-value
s should be the target column.
# Further, we don't need the Id or idhogar columsn because they are lablels not made for peforming
 fitting. They are categorical data and we should remove them.
X = trainDF.drop(["Id", "idhogar", "Target"],axis=1)
```

In [168]:
```
# Some of the data points are yes and no. Here, yes = 1 and no = 0. We must convert the data.
X = X.replace(to_replace = ["no", "yes"], value = [0, 1])
```

In [169]:
```
# We must fill all the NaN values in the dataset. First, determine which columns have NaN values.

for column in X.columns:
    if(X[column].isna().any() == True):
        print(column)
```

```
v2a1
v18q1
rez_esc
meaneduc
SQBmeaned
```

In [170]:
```
# v2a1 is monthly rent payment and is NaN when rent is not needed. We can put zero here.
# V18q1 is NaN when no tablets are owned, so we put 0. rez_esc is NaN when the individual is not be
hind in school. Put 0.
# meaneduc is NaN when the individual has no education. Put 0.
# SQBmeaned is square of  meaneduc, so put 0.
# All NaN values will become zero
X.fillna(value = 0.0, inplace = True)
```

In [171]:
```
X.isna().any()
```

Out[171]:
```
v2a1               False
hacdor             False
rooms              False
hacapo             False
v14a               False
                   ...
SQBhogar_nin       False
SQBovercrowding    False
SQBdependency      False
SQBmeaned          False
agesq              False
Length: 140, dtype: bool
```

In [172]:
```
X.T
```

Out[172]:

	0	1	2	3	4	5	6	7	8	9	...	
v2a1	190000.0	135000.0	0.0	180000.0	180000.0	180000.0	180000.0	130000.0	130000.0	130000.0	...	4
hacdor	0	0	0	0	0	0	0	1	1	1	...	
rooms	3	4	8	5	5	5	5	2	2	2	...	
hacapo	0	0	0	0	0	0	0	0	0	0	...	
v14a	1	1	1	1	1	1	1	1	1	1	...	
...	...	...	...	...	...	...	...	...	...	...	...	
SQBhogar_nin	0	0	0	4	4	4	4	4	4	4	...	
SQBovercrowding	1.0	1.0	0.25	1.777778	1.777778	1.777778	1.777778	16.0	16.0	16.0	...	5.
SQBdependency	0.0	64.0	64.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	...	0.
SQBmeaned	100.0	144.0	121.0	121.0	121.0	121.0	121.0	100.0	100.0	100.0	...	
agesq	1849	4489	8464	289	1369	1444	64	49	900	784	...	

140 rows × 9557 columns

In [173]:
```
# Make y the target column and perform train test split.
y = trainDF['Target']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=101)
```

In [174]:
```
# Fit the random forest model to the data
rfc = RandomForestClassifier(n_estimators=600)
rfc.fit(X_train,y_train)
```

Out[174]:
```
RandomForestClassifier
RandomForestClassifier(n_estimators=600)
```

In [175]:
```
# Make predictions based on the model
predictions = rfc.predict(X_test)
from sklearn.metrics import classification_report,confusion_matrix
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

           1       0.93      0.75      0.83       234
           2       0.90      0.80      0.85       484
           3       0.96      0.67      0.79       372
           4       0.89      0.99      0.93      1778

    accuracy                           0.90      2868
   macro avg       0.92      0.80      0.85      2868
weighted avg       0.90      0.90      0.89      2868
```

In [176]:
```
# Get the accuracy score
from sklearn import metrics
accuracy = metrics.accuracy_score(y_test, predictions)
accuracy
```

Out[176]: 0.8971408647140865

Check the accuracy using random forest with cross validation

In [177]:
```
# Import the k-fold library
from sklearn.model_selection import KFold
```

In [178]:
```
kf = KFold(n_splits = 10, shuffle = True)
```

In [179]:
```
# Define the input and output data. It will be the same as before.
dataInput = X
dataOutput = y
```

In [180]:
```
for trainSet, testSet in kf.split(dataInput):
    print(trainSet, testSet)
```

```
[   0    1    2 ... 9553 9555 9556] [   4   20   28   33   35   48   62   68   75   88  104  106
112  130
  137  148  173  185  197  222  226  228  243  252  270  301  306  318
  324  338  346  348  352  354  357  359  364  367  376  400  406  411
  424  426  428  445  447  449  455  457  465  467  488  507  522  523
  524  537  538  539  548  552  557  559  586  589  608  629  644  663
  670  687  698  706  717  727  735  740  747  750  769  770  833  846
  850  870  891  899  915  922  928  966  972 1009 1028 1031 1057 1058
 1077 1088 1098 1111 1122 1135 1160 1174 1185 1190 1194 1195 1209 1220
 1243 1256 1257 1259 1261 1274 1283 1288 1300 1304 1308 1312 1316 1321
 1347 1348 1350 1359 1364 1365 1393 1396 1410 1412 1413 1439 1441 1461
 1482 1487 1490 1509 1511 1516 1531 1541 1544 1545 1551 1559 1566 1575
 1587 1589 1597 1616 1619 1642 1683 1688 1706 1711 1721 1725 1732 1747
 1748 1753 1770 1794 1797 1803 1811 1812 1856 1872 1881 1892 1896 1911
 1925 1930 1936 1944 1954 1956 1972 1982 2002 2006 2015 2020 2033 2051
 2057 2069 2071 2075 2097 2105 2112 2113 2114 2120 2127 2135 2142 2145
 2149 2159 2175 2200 2206 2215 2220 2224 2227 2233 2234 2238 2257 2260
 2261 2266 2286 2320 2336 2340 2364 2372 2379 2382 2399 2417 2446 2450
 2489 2491 2498 2512 2514 2524 2540 2541 2547 2549 2551 2552 2556 2575
 2618 2625 2633 2654 2658 2659 2665 2667 2669 2677 2681 2706 2715 2740
 2746 2747 2751 2753 2759 2763 2769 2773 2775 2785 2788 2799 2831 2838
 2841 2849 2860 2872 2874 2892 2902 2940 2943 2945 2962 2966 2996 3012
 3017 3020 3022 3030 3044 3046 3047 3071 3098 3115 3129 3134 3136 3154
 3163 3169 3173 3176 3181 3192 3194 3195 3205 3206 3223 3233 3238 3241
 3244 3258 3266 3277 3279 3282 3296 3301 3303 3316 3326 3327 3330 3337
 3359 3371 3376 3383 3384 3400 3406 3407 3410 3412 3414 3417 3420 3433
 3436 3458 3466 3491 3501 3507 3535 3538 3549 3557 3560 3564 3569 3574
 3578 3581 3585 3588 3589 3630 3634 3635 3636 3637 3649 3656 3659 3686
 3706 3718 3719 3735 3736 3742 3747 3778 3795 3820 3825 3826 3828 3861
 3870 3876 3880 3899 3901 3905 3919 3923 3925 3928 3929 3935 3939 3957
 3960 3966 3985 3989 3990 3996 4008 4021 4024 4028 4052 4058 4076 4091
 4101 4107 4118 4119 4120 4152 4161 4166 4167 4175 4178 4188 4220 4226
 4246 4263 4268 4276 4277 4279 4294 4315 4318 4323 4334 4344 4360 4392
 4400 4417 4426 4437 4439 4444 4457 4475 4498 4505 4518 4531 4532 4551
 4575 4601 4606 4617 4645 4661 4662 4672 4681 4684 4720 4730 4745 4758
 4763 4769 4791 4798 4806 4808 4823 4828 4838 4850 4851 4856 4866 4913
 4932 4933 4935 4938 4943 4949 4969 4980 4987 4994 5018 5021 5025 5036
 5050 5059 5127 5130 5140 5145 5152 5169 5196 5223 5229 5232 5235 5246
 5249 5279 5288 5290 5294 5296 5313 5321 5324 5332 5359 5360 5383 5400
 5405 5410 5424 5428 5442 5449 5465 5473 5475 5483 5489 5495 5504 5511
 5517 5518 5528 5531 5555 5572 5584 5593 5601 5608 5611 5613 5619 5621
 5627 5630 5642 5645 5647 5655 5662 5663 5687 5689 5692 5697 5701 5703
 5704 5706 5720 5726 5746 5784 5790 5814 5821 5837 5860 5873 5877 5890
 5898 5901 5919 5931 5937 5975 5997 6000 6001 6002 6005 6011 6013 6014
 6016 6027 6032 6036 6050 6063 6065 6073 6077 6088 6091 6095 6099 6102
 6113 6122 6131 6142 6153 6158 6161 6179 6180 6184 6191 6202 6223 6225
 6226 6227 6228 6236 6255 6264 6265 6276 6288 6344 6360 6372 6376 6377
 6385 6406 6413 6447 6460 6468 6480 6504 6505 6508 6509 6510 6512 6535
 6547 6558 6559 6565 6567 6599 6633 6664 6682 6683 6685 6694 6700 6708
 6752 6757 6766 6787 6789 6812 6820 6830 6832 6833 6852 6858 6876 6888
 6912 6917 6918 6925 6931 6941 6956 6958 6961 6962 6989 7002 7008 7010
 7036 7055 7061 7066 7078 7084 7091 7103 7110 7115 7116 7136 7141 7145
 7151 7166 7177 7178 7196 7208 7222 7227 7230 7231 7234 7240 7306 7323
 7327 7328 7337 7353 7373 7375 7385 7394 7402 7411 7417 7421 7435 7441
 7453 7457 7464 7477 7484 7490 7497 7501 7529 7538 7546 7550 7551 7552
 7557 7561 7564 7567 7576 7597 7609 7613 7624 7636 7667 7684 7687 7717
 7733 7769 7773 7800 7805 7831 7839 7844 7846 7852 7865 7873 7880 7887
 7918 7932 7938 7948 7960 7975 7976 7980 7989 7991 8000 8014 8020 8023
 8030 8053 8062 8072 8076 8083 8120 8122 8127 8130 8146 8149 8150 8157
 8158 8159 8162 8165 8184 8188 8205 8214 8218 8225 8232 8233 8239 8240
 8244 8270 8283 8293 8300 8315 8323 8329 8333 8336 8355 8356 8361 8370
 8372 8376 8377 8387 8402 8409 8413 8440 8456 8465 8480 8497 8540 8545
 8547 8549 8553 8556 8558 8561 8582 8585 8586 8598 8605 8611 8612 8614
 8648 8653 8662 8675 8677 8694 8696 8702 8723 8726 8731 8740 8743 8758
 8759 8769 8784 8805 8817 8824 8826 8835 8843 8858 8867 8874 8896 8910
 8912 8918 8933 8942 8946 8950 8976 9012 9013 9017 9036 9048 9071 9072
 9079 9088 9091 9104 9118 9120 9122 9125 9127 9128 9129 9132 9133 9140
 9166 9171 9173 9179 9182 9205 9228 9259 9280 9296 9297 9310 9331 9347
 9355 9358 9359 9376 9379 9390 9393 9421 9424 9450 9464 9466 9480 9487
```

```
 9494 9519 9523 9554]
[   0    1    2 ... 9554 9555 9556] [   7   12   52   80   84   87   91   93  116  119  138  144
157  165
  172  179  203  208  233  240  249  255  260  274  302  340  344  353
  355  361  375  388  394  437  444  456  462  469  475  478  480  485
  498  511  514  516  567  569  573  576  585  600  612  616  645  651
  656  664  672  680  691  697  700  709  742  760  771  779  791  797
  803  807  810  812  814  843  847  861  862  864  865  871  890  892
  909  914  918  921  924  930  931  952  956  963  968  970  980  982
  985 1005 1013 1025 1045 1048 1053 1061 1066 1074 1083 1087 1129 1132
 1140 1144 1170 1172 1176 1205 1207 1221 1246 1277 1326 1333 1335 1355
 1361 1363 1368 1369 1375 1380 1387 1389 1394 1406 1409 1415 1431 1432
 1440 1457 1472 1498 1500 1508 1522 1532 1533 1536 1542 1572 1573 1598
 1613 1627 1640 1655 1659 1679 1680 1681 1686 1687 1696 1712 1717 1724
 1726 1727 1742 1761 1763 1766 1782 1784 1785 1799 1806 1808 1813 1821
 1824 1835 1854 1870 1878 1882 1883 1889 1898 1919 1924 1946 1958 1961
 1979 1980 1987 1988 1990 2010 2036 2042 2048 2059 2076 2084 2103 2110
 2123 2124 2130 2154 2156 2161 2164 2198 2202 2208 2214 2235 2239 2241
 2265 2274 2275 2280 2296 2300 2312 2321 2328 2332 2341 2345 2347 2349
 2366 2369 2388 2390 2394 2410 2415 2429 2430 2437 2439 2440 2453 2460
 2468 2490 2503 2518 2532 2548 2565 2595 2600 2617 2623 2624 2641 2646
 2656 2661 2668 2684 2691 2694 2703 2718 2724 2725 2734 2739 2770 2774
 2777 2779 2798 2810 2814 2816 2835 2856 2877 2916 2919 2936 2980 2993
 3000 3009 3024 3051 3056 3060 3093 3105 3108 3109 3122 3137 3139 3161
 3189 3203 3209 3230 3234 3237 3259 3313 3321 3325 3328 3332 3334 3347
 3367 3382 3385 3396 3398 3413 3422 3423 3431 3451 3461 3465 3470 3474
 3490 3494 3496 3500 3503 3506 3523 3524 3529 3532 3539 3546 3548 3573
 3582 3599 3600 3624 3626 3643 3644 3653 3661 3683 3695 3699 3707 3708
 3720 3726 3741 3752 3762 3764 3766 3787 3791 3799 3809 3811 3829 3835
 3838 3859 3882 3888 3908 3912 3918 3921 3955 3956 3961 3971 3973 3980
 3986 3995 4007 4020 4037 4059 4063 4072 4083 4087 4096 4110 4137 4147
 4155 4157 4168 4170 4171 4173 4177 4185 4212 4224 4234 4236 4238 4243
 4250 4252 4256 4259 4261 4283 4293 4308 4345 4347 4359 4373 4375 4395
 4399 4401 4440 4445 4446 4450 4451 4452 4464 4466 4477 4484 4493 4502
 4525 4529 4538 4554 4579 4590 4594 4609 4612 4630 4653 4660 4687 4688
 4691 4692 4705 4709 4715 4717 4721 4732 4751 4753 4764 4802 4809 4839
 4869 4871 4880 4881 4888 4909 4917 4918 4936 4941 4950 4977 4982 4996
 4998 5000 5016 5020 5044 5060 5062 5093 5098 5103 5109 5117 5119 5128
 5133 5146 5160 5162 5170 5174 5195 5216 5218 5234 5238 5243 5248 5254
 5257 5258 5272 5276 5278 5289 5293 5298 5304 5336 5347 5353 5361 5365
 5373 5380 5385 5390 5406 5419 5426 5429 5433 5458 5471 5472 5481 5503
 5513 5527 5552 5568 5575 5579 5585 5586 5587 5600 5618 5623 5643 5649
 5654 5658 5661 5676 5708 5752 5754 5765 5769 5798 5806 5822 5825 5831
 5833 5835 5842 5850 5876 5912 5917 5938 5940 5941 5970 5980 5998 6004
 6018 6025 6029 6033 6041 6049 6051 6067 6110 6119 6128 6136 6138 6140
 6155 6160 6165 6195 6205 6215 6222 6238 6241 6242 6248 6251 6262 6268
 6274 6285 6297 6301 6304 6312 6314 6329 6350 6362 6364 6368 6386 6395
 6405 6412 6422 6428 6448 6487 6495 6520 6525 6534 6539 6545 6546 6553
 6594 6604 6606 6612 6623 6629 6646 6655 6657 6663 6713 6717 6724 6727
 6729 6736 6741 6747 6749 6801 6811 6821 6860 6873 6880 6881 6884 6891
 6906 6916 6938 6940 6955 6978 6981 6983 6985 6991 6994 6995 7000 7013
 7019 7024 7026 7032 7033 7039 7042 7049 7063 7073 7074 7088 7092 7098
 7112 7122 7130 7146 7148 7156 7157 7158 7176 7184 7202 7221 7223 7232
 7233 7235 7238 7242 7249 7256 7258 7266 7272 7274 7294 7303 7304 7305
 7317 7329 7330 7339 7345 7354 7362 7374 7393 7400 7403 7406 7425 7433
 7440 7460 7463 7471 7472 7473 7483 7502 7541 7547 7554 7555 7582 7583
 7590 7606 7610 7611 7621 7627 7635 7640 7652 7653 7655 7672 7699 7707
 7708 7711 7729 7738 7753 7755 7756 7759 7765 7772 7778 7786 7812 7814
 7822 7826 7829 7843 7859 7869 7872 7882 7901 7923 7942 7945 7947 7949
 7958 7992 7998 8003 8004 8007 8010 8012 8019 8028 8033 8041 8054 8061
 8074 8088 8092 8104 8112 8113 8118 8138 8147 8161 8181 8199 8204 8217
 8230 8242 8243 8253 8256 8265 8266 8288 8305 8367 8379 8383 8397 8404
 8405 8412 8435 8442 8452 8455 8474 8489 8501 8522 8531 8532 8544 8548
 8557 8560 8576 8590 8609 8625 8630 8635 8636 8655 8664 8672 8674 8680
 8684 8689 8695 8698 8703 8711 8732 8733 8760 8764 8765 8780 8789 8796
 8819 8859 8865 8866 8868 8881 8895 8898 8905 8936 8938 8951 8957 8983
 8984 8994 9032 9039 9043 9045 9046 9062 9080 9081 9102 9103 9116 9121
 9123 9131 9137 9150 9174 9183 9186 9188 9189 9195 9196 9230 9241 9242
 9266 9278 9279 9291 9293 9308 9329 9356 9357 9363 9366 9367 9370 9382
 9385 9408 9423 9439 9442 9445 9452 9460 9463 9469 9470 9474 9497 9514
```

```
 9515 9526 9531 9542]
[   0    1    3 ... 9554 9555 9556] [   2   10   13   15   19   23   26   30   40   46   58   72
81   82
   85   95  100  105  128  129  143  158  166  174  186  200  202  206
  209  230  232  234  235  239  247  253  257  262  278  293  304  309
  328  330  339  350  380  382  413  420  423  429  450  459  472  473
  483  484  489  490  492  494  508  518  527  530  577  584  607  613
  626  636  642  647  648  652  657  662  666  701  718  732  739  745
  753  754  755  758  761  768  785  787  799  806  808  809  823  845
  849  851  877  879  902  916  923  935  936  947  957  977  978  989
  991 1001 1004 1018 1021 1037 1047 1049 1078 1080 1085 1090 1091 1109
 1168 1204 1206 1215 1222 1245 1250 1252 1262 1266 1268 1272 1292 1296
 1297 1322 1330 1352 1354 1373 1379 1397 1404 1411 1419 1429 1447 1450
 1451 1459 1467 1471 1483 1486 1495 1503 1505 1520 1521 1526 1527 1529
 1534 1560 1577 1585 1592 1600 1601 1609 1618 1639 1650 1667 1698 1700
 1718 1738 1749 1762 1764 1765 1772 1774 1789 1790 1791 1793 1801 1830
 1837 1850 1861 1877 1879 1880 1895 1904 1909 1921 1931 1945 1974 1977
 1984 1985 2008 2011 2014 2021 2041 2043 2047 2053 2061 2063 2080 2089
 2091 2093 2122 2160 2162 2166 2167 2168 2173 2176 2180 2187 2190 2192
 2213 2218 2230 2243 2255 2271 2283 2284 2298 2303 2311 2329 2350 2357
 2359 2377 2391 2392 2413 2424 2427 2449 2451 2461 2474 2488 2492 2510
 2534 2543 2555 2564 2572 2574 2582 2596 2597 2598 2610 2613 2621 2626
 2647 2653 2675 2678 2698 2709 2711 2712 2776 2778 2781 2787 2792 2808
 2813 2826 2827 2861 2868 2884 2888 2909 2925 2928 2932 2947 2959 2969
 2971 2975 2981 2988 2990 3006 3008 3016 3019 3029 3038 3039 3059 3061
 3088 3110 3113 3128 3148 3155 3170 3178 3199 3213 3214 3219 3228 3261
 3265 3269 3270 3276 3280 3284 3302 3310 3317 3318 3339 3340 3351 3352
 3363 3364 3365 3373 3403 3408 3411 3424 3439 3473 3475 3482 3492 3493
 3512 3516 3517 3533 3545 3554 3555 3562 3587 3590 3598 3603 3613 3617
 3619 3639 3664 3681 3696 3709 3716 3734 3739 3740 3761 3777 3786 3802
 3805 3818 3821 3832 3833 3839 3844 3847 3866 3889 3894 3944 3950 3965
 3967 3974 3976 3977 3979 3991 3993 4002 4016 4035 4066 4067 4068 4078
 4082 4099 4106 4128 4130 4140 4149 4174 4181 4198 4200 4201 4204 4208
 4209 4214 4228 4231 4239 4264 4269 4274 4284 4310 4340 4346 4361 4365
 4382 4384 4394 4412 4429 4432 4442 4458 4461 4474 4480 4487 4489 4509
 4510 4526 4528 4541 4543 4561 4569 4570 4571 4573 4578 4581 4593 4596
 4603 4622 4623 4626 4644 4649 4650 4654 4658 4668 4670 4673 4675 4676
 4694 4702 4727 4739 4741 4755 4756 4765 4767 4782 4794 4796 4830 4836
 4846 4852 4874 4879 4890 4892 4912 4937 4945 4948 4951 4952 4964 4991
 4992 5041 5080 5086 5112 5123 5125 5137 5148 5158 5172 5181 5191 5208
 5241 5251 5253 5260 5266 5269 5286 5301 5310 5315 5318 5343 5350 5356
 5363 5366 5388 5389 5391 5397 5399 5415 5448 5451 5500 5502 5508 5523
 5533 5546 5557 5566 5580 5589 5597 5605 5615 5626 5629 5631 5636 5653
 5664 5667 5668 5672 5673 5674 5686 5695 5698 5715 5718 5723 5729 5760
 5766 5770 5772 5792 5793 5804 5810 5815 5820 5841 5861 5882 5891 5915
 5927 5948 5949 5950 5964 5966 5969 5971 5976 5983 5989 6008 6009 6017
 6023 6031 6044 6053 6056 6069 6089 6096 6123 6147 6148 6159 6162 6163
 6169 6175 6181 6194 6206 6224 6243 6249 6257 6259 6277 6282 6287 6292
 6295 6308 6310 6321 6326 6331 6333 6340 6346 6352 6353 6355 6369 6398
 6409 6418 6440 6442 6456 6463 6465 6474 6486 6488 6493 6502 6533 6538
 6540 6541 6542 6543 6548 6561 6562 6563 6581 6600 6628 6649 6651 6652
 6687 6691 6707 6710 6711 6712 6733 6742 6745 6746 6771 6774 6778 6780
 6804 6815 6829 6835 6837 6838 6855 6877 6878 6883 6903 6919 6923 6942
 6950 6952 6953 6970 6988 6997 7014 7016 7018 7027 7052 7058 7079 7111
 7114 7123 7126 7133 7175 7182 7192 7204 7207 7209 7214 7216 7236 7239
 7254 7257 7279 7288 7292 7301 7322 7338 7340 7342 7343 7350 7378 7388
 7398 7404 7405 7414 7418 7437 7444 7446 7454 7467 7474 7485 7487 7506
 7525 7526 7553 7563 7577 7592 7603 7615 7620 7631 7633 7654 7666 7673
 7685 7698 7703 7714 7716 7730 7742 7748 7757 7761 7787 7790 7794 7823
 7835 7868 7883 7907 7936 7937 7959 7961 7977 7993 7994 7999 8009 8022
 8035 8042 8066 8095 8105 8119 8128 8137 8143 8151 8168 8179 8202 8203
 8206 8208 8228 8235 8236 8245 8262 8271 8272 8281 8310 8311 8314 8316
 8317 8330 8335 8338 8345 8349 8366 8371 8374 8375 8382 8386 8401 8406
 8408 8422 8427 8430 8441 8449 8457 8463 8464 8467 8498 8519 8521 8523
 8529 8530 8543 8552 8568 8571 8578 8587 8591 8603 8610 8634 8639 8650
 8668 8685 8686 8705 8724 8730 8734 8748 8750 8772 8776 8785 8795 8799
 8803 8822 8829 8851 8862 8870 8880 8903 8907 8908 8915 8919 8920 8968
 8972 8974 8985 8998 9000 9008 9009 9014 9025 9026 9029 9052 9064 9084
 9095 9117 9136 9184 9194 9251 9277 9292 9299 9301 9307 9309 9340 9343
 9365 9371 9373 9397 9402 9406 9407 9409 9412 9416 9417 9446 9449 9457
```

```
 9508 9532 9546 9552]
[   1    2    4 ... 9553 9554 9555] [   0    3    8   34   43   54   61   63   73   94   97  110
115  117
  127  145  146  149  159  189  193  195  196  201  207  210  264  265
  288  297  303  308  327  331  337  351  362  393  398  405  407  408
  414  421  438  470  477  486  500  502  504  505  506  512  536  540
  550  553  564  596  603  628  631  650  655  658  660  676  681  682
  686  705  725  764  773  788  790  793  798  805  815  825  831  839
  841  842  848  859  881  905  911  912  929  933  938  939  948  949
  965  979  990  996  998 1007 1008 1010 1041 1050 1063 1075 1079 1081
 1082 1097 1102 1105 1110 1126 1159 1164 1169 1175 1181 1184 1191 1196
 1198 1201 1212 1224 1225 1227 1229 1241 1254 1255 1279 1285 1291 1295
 1302 1306 1317 1328 1336 1339 1351 1356 1370 1386 1391 1400 1401 1425
 1430 1435 1465 1468 1473 1474 1502 1525 1543 1579 1583 1607 1630 1632
 1634 1677 1690 1692 1716 1722 1731 1735 1737 1740 1746 1836 1841 1844
 1846 1858 1885 1894 1900 1914 1915 1917 1940 1941 1953 1968 1973 1981
 2003 2007 2025 2054 2055 2062 2070 2079 2094 2100 2131 2144 2151 2153
 2170 2177 2181 2195 2196 2209 2216 2221 2231 2242 2245 2248 2251 2263
 2278 2295 2326 2338 2343 2346 2351 2356 2365 2370 2374 2378 2396 2398
 2403 2405 2414 2425 2434 2443 2459 2462 2494 2495 2502 2504 2507 2513
 2526 2530 2567 2569 2577 2587 2589 2599 2607 2629 2644 2650 2657 2672
 2674 2679 2685 2690 2693 2695 2701 2707 2708 2782 2802 2809 2811 2825
 2828 2839 2854 2858 2859 2870 2878 2904 2929 2938 2949 2955 2968 2970
 2976 2978 2979 2983 2984 2989 2991 2994 3013 3023 3025 3026 3036 3043
 3045 3050 3064 3080 3099 3117 3125 3131 3142 3145 3157 3180 3200 3240
 3245 3246 3247 3263 3268 3281 3289 3290 3291 3294 3297 3306 3308 3315
 3333 3344 3358 3392 3394 3404 3430 3445 3450 3477 3485 3515 3527 3540
 3541 3552 3556 3558 3607 3614 3622 3628 3665 3667 3682 3688 3693 3704
 3705 3713 3725 3755 3756 3767 3788 3790 3796 3797 3800 3807 3822 3837
 3840 3846 3864 3879 3883 3887 3892 3906 3933 3962 4015 4022 4027 4033
 4044 4049 4055 4056 4060 4100 4102 4112 4126 4141 4143 4154 4156 4172
 4193 4195 4202 4206 4215 4242 4251 4253 4257 4262 4290 4292 4307 4331
 4333 4336 4338 4358 4367 4369 4379 4381 4386 4390 4406 4407 4411 4413
 4414 4420 4430 4433 4470 4491 4517 4521 4533 4537 4549 4558 4560 4568
 4576 4584 4588 4598 4600 4605 4607 4610 4611 4618 4625 4646 4679 4693
 4707 4708 4712 4723 4724 4734 4750 4759 4768 4771 4772 4789 4800 4811
 4814 4841 4844 4870 4885 4887 4901 4908 4915 4922 4929 4934 4944 4971
 4984 5004 5009 5010 5017 5019 5029 5033 5038 5040 5042 5048 5073 5078
 5088 5097 5107 5118 5120 5143 5154 5167 5184 5186 5192 5212 5221 5226
 5236 5242 5306 5312 5319 5337 5379 5395 5404 5407 5413 5423 5431 5447
 5455 5456 5460 5469 5484 5486 5491 5492 5507 5516 5519 5522 5525 5536
 5537 5538 5539 5542 5548 5554 5558 5560 5581 5609 5628 5641 5644 5651
 5657 5659 5660 5680 5681 5700 5719 5725 5735 5736 5740 5756 5762 5763
 5779 5783 5794 5795 5829 5832 5843 5848 5864 5874 5892 5907 5911 5914
 5916 5921 5925 5930 5946 5951 5955 5957 5984 6024 6038 6042 6071 6074
 6078 6090 6112 6114 6120 6121 6129 6133 6144 6150 6151 6185 6188 6209
 6213 6217 6244 6254 6258 6280 6296 6300 6313 6319 6341 6347 6351 6359
 6373 6374 6384 6394 6400 6404 6430 6435 6450 6466 6470 6471 6472 6475
 6477 6490 6498 6501 6507 6517 6555 6560 6566 6575 6577 6586 6591 6601
 6603 6607 6676 6679 6681 6697 6722 6725 6730 6731 6732 6740 6783 6785
 6786 6793 6808 6814 6831 6834 6845 6874 6887 6920 6924 6944 6946 6968
 6969 6971 6980 6982 6998 7012 7021 7028 7054 7056 7059 7071 7072 7076
 7080 7087 7104 7128 7153 7168 7171 7181 7205 7226 7229 7243 7268 7290
 7295 7297 7309 7310 7314 7332 7341 7347 7348 7355 7357 7361 7368 7370
 7408 7410 7424 7431 7436 7439 7445 7455 7459 7476 7486 7510 7544 7545
 7558 7562 7568 7574 7575 7578 7593 7605 7607 7614 7619 7625 7647 7662
 7664 7680 7691 7695 7712 7731 7732 7763 7764 7782 7783 7785 7798 7801
 7815 7816 7819 7864 7881 7890 7902 7904 7919 7922 7941 7956 7964 7974
 7996 8002 8005 8013 8018 8021 8024 8025 8038 8046 8065 8073 8078 8082
 8089 8094 8097 8107 8111 8117 8133 8170 8185 8192 8196 8198 8213 8216
 8223 8251 8259 8264 8268 8275 8280 8282 8285 8296 8318 8322 8340 8381
 8398 8417 8423 8424 8433 8438 8443 8444 8446 8451 8454 8466 8471 8477
 8512 8517 8520 8536 8537 8538 8539 8554 8574 8575 8580 8595 8597 8600
 8613 8621 8632 8641 8654 8663 8667 8676 8690 8700 8701 8747 8751 8770
 8774 8775 8779 8782 8786 8791 8793 8825 8831 8838 8841 8849 8852 8857
 8872 8878 8882 8890 8897 8909 8917 8926 8932 8939 8977 8978 8986 8991
 8996 9001 9010 9015 9021 9022 9033 9042 9044 9065 9066 9069 9083 9107
 9109 9111 9115 9141 9167 9169 9178 9191 9204 9218 9225 9249 9258 9260
 9261 9268 9276 9283 9284 9312 9313 9319 9323 9332 9344 9350 9352 9354
 9362 9364 9368 9394 9399 9415 9427 9435 9448 9451 9454 9479 9504 9507
```

```
 9517 9518 9520 9556]
[   0    1    2 ... 9554 9555 9556] [   9   14   22   24   39   60   70   78  108  126  131  135
136  151
  152  156  164  177  181  198  199  211  212  217  220  241  250  259
  266  267  273  280  284  287  292  298  315  341  347  358  371  372
  379  391  396  417  440  442  443  461  474  487  493  495  499  509
  535  545  554  560  562  566  570  580  587  593  598  601  605  609
  610  617  635  659  669  673  693  714  721  734  736  759  772  774
  786  789  794  795  819  824  826  837  856  869  873  883  889  900
  908  926  946  951  953  964  975  983  984  987  988 1000 1017 1019
 1032 1044 1068 1070 1072 1096 1118 1138 1141 1143 1147 1148 1154 1156
 1161 1192 1197 1210 1232 1238 1269 1270 1271 1273 1281 1315 1319 1324
 1329 1344 1349 1358 1378 1395 1403 1407 1445 1469 1470 1478 1479 1484
 1485 1488 1515 1523 1540 1562 1578 1599 1612 1621 1624 1635 1636 1641
 1645 1662 1666 1669 1691 1699 1702 1704 1705 1707 1709 1728 1730 1734
 1736 1752 1758 1809 1822 1826 1831 1839 1843 1853 1867 1868 1874 1907
 1922 1952 1957 1962 1967 1976 1978 1989 1997 2001 2017 2030 2046 2060
 2085 2086 2090 2096 2102 2118 2126 2128 2147 2157 2178 2184 2197 2205
 2207 2236 2247 2258 2279 2304 2306 2316 2317 2319 2360 2368 2395 2397
 2411 2433 2435 2456 2464 2472 2496 2520 2533 2535 2542 2544 2554 2566
 2570 2590 2605 2615 2619 2620 2622 2627 2628 2638 2649 2666 2702 2713
 2723 2726 2730 2731 2732 2738 2741 2745 2755 2758 2762 2771 2796 2800
 2805 2824 2837 2847 2850 2864 2867 2871 2894 2896 2897 2903 2908 2910
 2911 2913 2914 2930 2946 3010 3031 3049 3052 3063 3084 3086 3094 3103
 3114 3119 3135 3140 3143 3149 3165 3167 3171 3193 3212 3218 3222 3248
 3300 3312 3349 3375 3380 3387 3395 3426 3435 3454 3456 3471 3472 3480
 3499 3504 3508 3514 3531 3536 3550 3563 3571 3618 3620 3623 3647 3651
 3652 3655 3658 3671 3672 3698 3700 3701 3727 3743 3753 3776 3779 3803
 3808 3812 3815 3817 3836 3845 3867 3872 3877 3917 3934 3937 3949 3951
 3959 3968 3975 3978 3981 3983 3987 4004 4009 4010 4011 4017 4025 4036
 4042 4043 4046 4047 4050 4054 4070 4084 4098 4117 4121 4127 4142 4151
 4187 4189 4196 4219 4229 4230 4233 4245 4255 4281 4289 4295 4304 4306
 4313 4326 4332 4355 4356 4366 4371 4374 4376 4388 4402 4410 4419 4441
 4449 4454 4459 4488 4501 4504 4553 4555 4556 4559 4567 4572 4582 4583
 4585 4592 4614 4616 4624 4639 4652 4656 4686 4689 4701 4713 4725 4733
 4737 4744 4747 4748 4754 4780 4783 4787 4799 4824 4825 4832 4848 4853
 4854 4858 4860 4875 4876 4884 4891 4898 4900 4905 4921 4923 4930 4939
 4946 4953 4957 4958 5005 5037 5053 5057 5058 5061 5066 5074 5089 5091
 5115 5131 5138 5149 5163 5175 5176 5183 5188 5200 5203 5214 5215 5224
 5227 5237 5247 5250 5259 5264 5265 5280 5281 5285 5305 5320 5325 5334
 5344 5352 5371 5374 5377 5381 5396 5398 5409 5411 5417 5420 5439 5440
 5444 5464 5477 5479 5487 5488 5499 5534 5565 5578 5582 5590 5606 5622
 5677 5688 5694 5699 5702 5721 5722 5727 5731 5741 5750 5753 5755 5768
 5773 5777 5780 5782 5789 5817 5839 5854 5855 5857 5863 5875 5881 5884
 5888 5895 5896 5904 5908 5909 5923 5924 5936 5952 5958 5973 5979 5982
 5985 5991 6010 6026 6054 6059 6064 6070 6072 6103 6141 6143 6170 6171
 6186 6190 6193 6216 6229 6232 6235 6281 6290 6291 6294 6371 6378 6391
 6392 6403 6408 6431 6436 6438 6445 6478 6499 6503 6506 6519 6528 6532
 6579 6584 6585 6617 6621 6626 6630 6637 6640 6643 6650 6658 6665 6670
 6689 6695 6705 6709 6720 6738 6753 6754 6760 6761 6767 6769 6781 6784
 6791 6797 6803 6810 6840 6846 6851 6900 6902 6905 6915 6922 6929 6933
 6939 6948 6951 6963 6966 6972 7007 7009 7038 7048 7057 7068 7077 7085
 7109 7113 7119 7132 7142 7143 7161 7162 7183 7189 7203 7212 7213 7218
 7219 7250 7260 7264 7270 7283 7311 7316 7319 7321 7331 7334 7336 7352
 7358 7369 7371 7382 7384 7386 7395 7399 7462 7478 7488 7489 7491 7493
 7498 7512 7515 7520 7523 7531 7549 7559 7566 7569 7570 7581 7585 7588
 7598 7608 7629 7660 7681 7683 7690 7693 7696 7700 7705 7709 7722 7727
 7728 7744 7747 7750 7751 7767 7791 7796 7817 7820 7827 7840 7849 7856
 7885 7891 7899 7925 7926 7940 7951 7953 7963 7965 7969 7986 8011 8016
 8032 8036 8040 8043 8057 8059 8067 8070 8085 8099 8100 8103 8129 8140
 8148 8166 8167 8174 8209 8212 8237 8284 8290 8301 8308 8350 8354 8359
 8363 8365 8384 8385 8395 8407 8415 8421 8426 8445 8448 8453 8460 8483
 8487 8488 8493 8503 8511 8533 8542 8570 8594 8601 8608 8627 8628 8629
 8637 8652 8656 8657 8658 8665 8669 8681 8682 8691 8706 8710 8715 8721
 8722 8752 8754 8755 8766 8787 8815 8828 8832 8845 8861 8864 8884 8899
 8901 8902 8906 8924 8943 8945 8948 8958 8963 8979 8987 8989 9004 9028
 9030 9035 9054 9055 9056 9075 9076 9082 9085 9101 9105 9106 9110 9112
 9126 9130 9154 9159 9161 9168 9170 9175 9177 9199 9202 9203 9207 9210
 9219 9222 9223 9235 9237 9244 9253 9255 9267 9294 9298 9311 9325 9342
 9372 9375 9391 9395 9400 9405 9419 9420 9429 9437 9444 9455 9462 9473
```

```
 9476 9505 9527 9528]
[   0    2    3 ... 9554 9555 9556] [   1   16   18   47   50   51   64   76  103  107  121  124
133  150
  154  161  162  175  180  194  214  219  223  237  242  248  254  272
  276  277  290  295  310  314  316  319  345  363  374  378  385  389
  409  422  430  433  434  448  451  501  517  526  534  542  544  546
  555  574  578  588  590  591  606  615  619  638  671  675  719  726
  729  730  738  746  766  778  783  784  796  802  830  835  840  867
  872  904  907  920  925  942  962  981 1011 1022 1030 1035 1051 1059
 1076 1095 1106 1116 1125 1127 1130 1142 1162 1163 1171 1182 1186 1189
 1208 1211 1218 1233 1235 1244 1263 1265 1275 1282 1294 1298 1301 1311
 1362 1374 1377 1381 1399 1405 1408 1416 1418 1424 1455 1458 1460 1462
 1463 1499 1510 1517 1528 1539 1547 1549 1554 1555 1563 1582 1588 1590
 1591 1611 1617 1629 1646 1651 1652 1653 1654 1663 1675 1676 1682 1695
 1708 1723 1750 1754 1757 1768 1771 1775 1777 1779 1783 1828 1832 1847
 1871 1893 1899 1912 1916 1923 1935 1937 1970 1983 1999 2000 2012 2022
 2027 2038 2049 2064 2081 2116 2134 2140 2143 2150 2169 2172 2174 2194
 2212 2240 2244 2250 2254 2267 2269 2285 2302 2308 2309 2310 2324 2330
 2333 2334 2342 2386 2387 2401 2406 2408 2409 2412 2422 2426 2428 2458
 2482 2484 2497 2499 2505 2509 2511 2531 2536 2545 2550 2553 2557 2563
 2591 2603 2608 2609 2616 2639 2640 2642 2651 2663 2671 2680 2705 2719
 2729 2733 2737 2748 2750 2766 2768 2783 2784 2786 2812 2822 2823 2852
 2853 2880 2882 2889 2900 2905 2920 2924 2933 2934 2937 2944 2950 2953
 2958 2961 2973 2974 2987 2997 3007 3021 3042 3048 3055 3058 3082 3087
 3100 3111 3112 3118 3127 3141 3146 3151 3153 3160 3166 3175 3184 3208
 3229 3232 3236 3250 3251 3252 3272 3275 3292 3295 3299 3305 3309 3311
 3322 3356 3362 3372 3377 3379 3390 3421 3432 3438 3446 3464 3476 3479
 3481 3483 3487 3519 3537 3566 3583 3594 3597 3602 3638 3640 3645 3648
 3670 3673 3675 3679 3729 3731 3732 3737 3751 3758 3759 3763 3769 3773
 3774 3775 3823 3852 3857 3884 3886 3891 3903 3910 3914 3915 3930 3941
 3954 3970 3984 4012 4018 4034 4048 4062 4065 4074 4081 4090 4111 4115
 4116 4122 4125 4129 4133 4136 4144 4148 4153 4182 4183 4186 4205 4210
 4244 4248 4267 4272 4275 4303 4312 4316 4317 4320 4322 4324 4342 4351
 4387 4408 4421 4424 4425 4434 4436 4455 4456 4465 4468 4472 4473 4481
 4490 4494 4495 4499 4500 4508 4511 4524 4530 4544 4545 4547 4563 4565
 4580 4620 4635 4642 4651 4655 4663 4685 4710 4714 4728 4757 4761 4776
 4803 4812 4817 4819 4826 4827 4831 4845 4849 4861 4895 4897 4914 4926
 4928 4942 4959 4966 4975 4983 4989 5001 5015 5030 5046 5047 5055 5083
 5084 5105 5114 5121 5126 5129 5144 5153 5157 5173 5187 5190 5194 5211
 5245 5256 5262 5267 5275 5303 5311 5322 5323 5327 5349 5362 5382 5386
 5412 5422 5427 5432 5446 5461 5467 5468 5485 5496 5510 5512 5515 5529
 5532 5535 5556 5588 5592 5595 5616 5632 5640 5665 5669 5683 5684 5696
 5709 5728 5730 5733 5747 5748 5759 5774 5775 5778 5791 5799 5801 5807
 5818 5824 5838 5846 5862 5872 5886 5897 5939 5962 5963 5965 5968 5977
 5981 5999 6003 6012 6020 6035 6040 6047 6061 6085 6087 6105 6125 6157
 6166 6167 6176 6183 6192 6204 6208 6219 6239 6263 6271 6272 6309 6316
 6320 6327 6332 6349 6390 6397 6415 6424 6426 6453 6473 6489 6494 6496
 6514 6515 6518 6521 6536 6551 6552 6569 6571 6590 6592 6602 6611 6636
 6639 6642 6644 6645 6654 6661 6675 6684 6701 6703 6704 6721 6739 6748
 6750 6759 6765 6788 6790 6792 6798 6807 6813 6819 6822 6824 6839 6848
 6863 6866 6870 6904 6936 6937 6945 6965 6984 6986 7005 7025 7029 7035
 7044 7046 7065 7075 7093 7102 7108 7118 7121 7137 7155 7163 7185 7191
 7194 7195 7197 7201 7211 7224 7244 7246 7247 7253 7273 7277 7281 7287
 7293 7298 7300 7308 7312 7333 7351 7365 7366 7390 7392 7409 7413 7415
 7434 7443 7458 7480 7481 7492 7494 7496 7507 7516 7517 7522 7530 7536
 7543 7548 7565 7594 7601 7602 7623 7638 7639 7641 7643 7645 7669 7678
 7682 7701 7704 7710 7725 7735 7737 7739 7740 7745 7774 7779 7788 7789
 7793 7799 7806 7818 7828 7838 7841 7847 7854 7855 7860 7861 7870 7878
 7886 7892 7898 7921 7927 7928 7939 7943 7950 7954 7972 7995 8001 8017
 8026 8039 8052 8056 8063 8064 8079 8091 8093 8098 8109 8125 8132 8144
 8154 8155 8169 8180 8195 8197 8221 8227 8231 8234 8248 8250 8252 8263
 8292 8306 8337 8341 8342 8352 8369 8380 8388 8392 8393 8400 8410 8416
 8420 8425 8428 8429 8431 8439 8447 8459 8470 8472 8484 8485 8486 8507
 8513 8516 8518 8528 8551 8563 8567 8577 8579 8581 8589 8593 8616 8624
 8640 8642 8645 8673 8687 8688 8739 8762 8767 8781 8797 8809 8812 8813
 8830 8854 8873 8875 8877 8887 8889 8916 8922 8929 8947 8954 8960 8990
 9023 9053 9061 9063 9067 9068 9077 9078 9092 9097 9134 9138 9147 9148
 9149 9163 9176 9187 9192 9200 9201 9229 9245 9250 9252 9262 9286 9287
 9290 9302 9314 9316 9322 9330 9335 9338 9345 9361 9378 9386 9389 9413
 9414 9418 9447 9456 9467 9478 9485 9488 9495 9502 9506 9524 9534 9535
```

```
 9537 9541 9544 9553]
[   0    1    2 ... 9553 9554 9556] [   5   21   27   38   56   59   65   96   98  101  114  120
132  155
  167  169  183  187  191  192  216  218  224  227  263  268  269  275
  281  289  296  299  300  323  334  365  368  369  373  397  399  401
  402  404  410  415  435  441  446  458  468  481  491  528  531  532
  533  547  565  571  572  592  621  630  640  641  654  678  679  683
  689  703  712  715  748  751  775  792  801  811  813  822  828  860
  863  875  878  888  896  917  932  937  944  950  971  973  997 1003
 1012 1014 1039 1046 1054 1055 1056 1062 1064 1065 1073 1084 1089 1092
 1094 1101 1113 1117 1119 1124 1131 1149 1166 1167 1178 1188 1203 1214
 1226 1228 1231 1234 1242 1247 1276 1284 1287 1289 1303 1307 1325 1334
 1420 1428 1433 1436 1464 1481 1494 1496 1497 1501 1512 1514 1537 1556
 1557 1558 1565 1569 1571 1576 1584 1594 1596 1604 1606 1608 1614 1626
 1631 1637 1638 1647 1649 1668 1671 1672 1693 1694 1697 1720 1743 1773
 1781 1815 1817 1819 1825 1842 1851 1866 1887 1906 1927 1929 1938 1948
 1950 1959 1965 1998 2004 2009 2016 2026 2031 2050 2068 2074 2077 2078
 2111 2119 2136 2138 2165 2186 2201 2211 2229 2256 2262 2276 2277 2287
 2291 2293 2318 2322 2344 2362 2363 2367 2373 2375 2385 2400 2438 2463
 2471 2486 2487 2515 2516 2523 2593 2614 2634 2645 2676 2682 2687 2689
 2704 2727 2752 2780 2804 2806 2820 2830 2832 2834 2843 2845 2851 2855
 2863 2875 2876 2881 2890 2891 2899 2906 2907 2912 2921 2927 2948 2952
 2956 2964 2965 3002 3004 3005 3015 3027 3053 3054 3066 3076 3077 3081
 3083 3091 3095 3101 3116 3121 3124 3156 3162 3164 3174 3196 3197 3201
 3202 3215 3220 3221 3225 3226 3249 3254 3255 3256 3257 3278 3307 3324
 3336 3338 3350 3369 3370 3393 3405 3419 3427 3441 3442 3444 3447 3449
 3457 3462 3467 3469 3489 3520 3530 3567 3577 3579 3584 3591 3606 3610
 3612 3625 3631 3657 3660 3674 3677 3684 3689 3692 3697 3721 3724 3730
 3745 3746 3768 3780 3781 3782 3783 3784 3801 3806 3843 3853 3854 3862
 3873 3904 3909 3926 3945 3946 3953 3963 3969 3972 3992 3999 4026 4061
 4075 4085 4088 4089 4093 4104 4113 4135 4146 4176 4179 4180 4221 4240
 4247 4249 4254 4260 4265 4273 4287 4296 4298 4299 4300 4302 4309 4319
 4321 4325 4327 4339 4352 4357 4370 4380 4391 4396 4398 4416 4418 4427
 4453 4469 4471 4483 4485 4486 4492 4496 4497 4503 4515 4527 4542 4550
 4587 4604 4632 4636 4640 4674 4680 4697 4700 4703 4704 4706 4711 4716
 4722 4729 4740 4742 4743 4762 4784 4785 4790 4793 4795 4797 4801 4807
 4821 4834 4840 4843 4847 4872 4886 4893 4896 4904 4956 4962 4981 4986
 4988 4993 4997 5008 5013 5039 5052 5064 5071 5075 5082 5104 5111 5113
 5132 5134 5142 5155 5156 5171 5177 5204 5205 5210 5220 5222 5231 5244
 5252 5284 5295 5297 5307 5308 5309 5316 5328 5333 5338 5340 5351 5354
 5357 5375 5401 5408 5418 5421 5425 5430 5434 5435 5437 5438 5441 5443
 5450 5452 5457 5463 5474 5476 5498 5501 5506 5509 5514 5541 5544 5545
 5549 5550 5563 5567 5569 5570 5583 5599 5604 5607 5612 5617 5637 5648
 5650 5652 5656 5666 5682 5690 5693 5707 5712 5713 5716 5724 5732 5738
 5749 5786 5800 5812 5823 5845 5866 5870 5871 5883 5905 5918 5920 5929
 5934 5943 5959 5992 5994 6037 6055 6060 6062 6080 6081 6083 6084 6092
 6097 6106 6117 6137 6154 6178 6199 6200 6203 6210 6211 6214 6221 6230
 6246 6253 6270 6305 6306 6315 6324 6325 6328 6335 6339 6354 6363 6379
 6380 6389 6410 6423 6437 6451 6454 6458 6479 6482 6484 6485 6491 6497
 6511 6513 6527 6531 6568 6573 6582 6588 6613 6618 6620 6622 6624 6641
 6647 6653 6667 6674 6692 6696 6714 6715 6728 6756 6758 6762 6764 6775
 6776 6777 6805 6818 6826 6827 6843 6850 6853 6854 6856 6859 6862 6867
 6871 6875 6879 6889 6895 6896 6897 6909 6926 6928 6932 6959 6973 6993
 6999 7015 7041 7051 7094 7099 7105 7107 7129 7134 7135 7138 7139 7149
 7164 7165 7170 7172 7198 7206 7217 7228 7251 7255 7262 7267 7269 7271
 7275 7286 7291 7302 7307 7325 7359 7367 7372 7389 7412 7420 7428 7429
 7430 7456 7466 7469 7479 7495 7503 7528 7535 7539 7556 7586 7587 7599
 7617 7622 7628 7632 7657 7671 7689 7692 7697 7743 7762 7792 7795 7813
 7824 7832 7833 7845 7850 7874 7875 7876 7877 7888 7915 7924 7934 7955
 7962 7966 7968 7982 7987 8015 8031 8048 8055 8075 8087 8101 8116 8121
 8126 8135 8145 8156 8183 8193 8194 8200 8211 8226 8258 8260 8269 8273
 8276 8286 8297 8298 8304 8312 8313 8320 8324 8331 8343 8357 8362 8394
 8418 8437 8468 8476 8491 8492 8506 8510 8515 8535 8546 8559 8562 8569
 8572 8573 8583 8584 8588 8592 8596 8602 8604 8606 8607 8617 8618 8622
 8626 8647 8649 8651 8660 8670 8679 8712 8713 8727 8729 8738 8742 8744
 8745 8792 8798 8801 8820 8836 8842 8848 8855 8856 8879 8888 8891 8904
 8911 8913 8921 8930 8937 8964 8965 8966 8981 8992 8993 8995 8999 9002
 9007 9020 9027 9037 9038 9049 9051 9059 9070 9094 9099 9100 9146 9153
 9162 9164 9172 9209 9212 9214 9221 9233 9234 9256 9264 9303 9304 9305
 9320 9321 9349 9387 9425 9428 9436 9441 9465 9477 9481 9491 9498 9501
```

```
 9516 9522 9539 9555]
[   0    1    2 ... 9554 9555 9556] [   6   32   36   37   42   49   53   79   89   99  109  123
125  147
  153  171  182  245  258  283  285  294  305  307  311  313  320  322
  329  332  333  335  343  349  366  384  392  427  431  432  439  454
  460  463  466  471  521  525  529  543  549  568  602  611  620  622
  624  625  627  632  634  637  661  665  668  690  692  694  710  711
  720  733  737  741  744  752  763  765  780  782  804  818  827  832
  844  854  857  858  866  868  880  882  886  887  895  910  919  941
  945  954  986  993 1015 1029 1036 1038 1040 1060 1067 1071 1093 1107
 1114 1128 1137 1146 1150 1151 1152 1155 1158 1173 1180 1183 1202 1219
 1223 1230 1237 1239 1260 1264 1290 1305 1309 1318 1323 1338 1340 1345
 1357 1367 1385 1392 1402 1417 1423 1426 1434 1437 1442 1444 1454 1476
 1477 1504 1518 1538 1550 1552 1561 1564 1568 1570 1574 1603 1615 1643
 1644 1665 1670 1678 1684 1703 1713 1714 1719 1729 1733 1744 1751 1755
 1776 1788 1798 1802 1805 1814 1818 1823 1834 1838 1849 1852 1857 1860
 1862 1864 1869 1875 1886 1891 1897 1901 1908 1910 1913 1920 1933 1934
 1949 1951 1966 1969 1991 1994 1995 2005 2024 2037 2044 2045 2058 2065
 2073 2082 2083 2088 2095 2098 2099 2101 2115 2121 2125 2163 2188 2189
 2193 2199 2204 2225 2226 2253 2270 2281 2282 2288 2292 2299 2305 2314
 2339 2354 2371 2383 2389 2418 2423 2436 2444 2448 2455 2466 2473 2493
 2500 2508 2517 2519 2522 2527 2529 2537 2546 2558 2560 2568 2571 2594
 2601 2606 2611 2635 2636 2648 2655 2660 2670 2688 2699 2710 2716 2721
 2722 2743 2772 2794 2797 2801 2815 2821 2829 2842 2844 2865 2869 2873
 2885 2895 2915 2917 2918 2923 2935 2954 2957 2963 2972 2986 2995 2998
 3001 3028 3034 3057 3065 3067 3068 3079 3085 3089 3120 3138 3144 3158
 3168 3172 3179 3182 3183 3188 3191 3216 3217 3271 3285 3286 3298 3319
 3331 3341 3342 3343 3345 3368 3374 3381 3388 3391 3397 3401 3415 3416
 3428 3429 3434 3440 3452 3455 3478 3484 3509 3518 3544 3551 3553 3561
 3572 3576 3580 3586 3632 3641 3650 3654 3666 3669 3676 3710 3711 3717
 3728 3733 3744 3749 3765 3770 3772 3792 3793 3798 3830 3831 3834 3841
 3849 3858 3885 3920 3931 3932 3940 3952 3997 3998 4006 4053 4057 4073
 4077 4109 4114 4123 4124 4132 4139 4150 4162 4184 4194 4199 4211 4217
 4218 4258 4270 4288 4301 4311 4328 4329 4341 4343 4354 4362 4377 4403
 4415 4422 4423 4431 4435 4448 4467 4516 4519 4534 4536 4548 4557 4562
 4577 4595 4602 4613 4621 4629 4641 4643 4647 4648 4657 4665 4669 4671
 4678 4683 4690 4719 4738 4770 4775 4804 4805 4816 4820 4842 4859 4865
 4878 4883 4889 4902 4906 4919 4920 4931 4954 4968 4972 4974 4976 4985
 4999 5003 5006 5011 5014 5022 5023 5028 5054 5067 5070 5095 5096 5102
 5108 5116 5124 5136 5151 5197 5198 5202 5209 5217 5239 5261 5282 5299
 5314 5326 5342 5345 5355 5358 5384 5394 5402 5416 5453 5466 5478 5480
 5482 5490 5494 5520 5524 5561 5576 5577 5602 5610 5614 5625 5646 5685
 5691 5717 5739 5742 5744 5757 5758 5761 5781 5787 5797 5802 5803 5809
 5827 5830 5840 5851 5856 5859 5885 5887 5889 5893 5899 5900 5910 5913
 5922 5953 5956 5960 5961 5967 5974 6006 6045 6052 6058 6068 6075 6100
 6101 6104 6108 6111 6115 6130 6168 6172 6174 6196 6197 6220 6231 6234
 6245 6250 6261 6267 6273 6284 6286 6298 6307 6317 6318 6322 6323 6334
 6338 6342 6345 6356 6357 6358 6370 6375 6387 6388 6396 6402 6414 6417
 6419 6429 6443 6449 6462 6464 6467 6481 6483 6492 6522 6524 6526 6529
 6554 6583 6589 6593 6595 6597 6598 6615 6619 6625 6631 6648 6662 6677
 6680 6690 6699 6706 6716 6718 6723 6726 6735 6743 6763 6779 6817 6849
 6857 6861 6864 6885 6892 6894 6899 6901 6913 6927 6930 6934 6947 6954
 6979 6990 6992 7003 7006 7011 7017 7031 7037 7045 7047 7053 7060 7081
 7082 7083 7086 7101 7120 7124 7127 7131 7140 7174 7180 7190 7199 7200
 7210 7241 7248 7252 7263 7278 7282 7284 7318 7320 7335 7344 7360 7377
 7379 7383 7407 7419 7422 7423 7438 7442 7447 7475 7499 7505 7509 7513
 7514 7518 7519 7534 7537 7591 7596 7612 7634 7644 7646 7661 7670 7674
 7675 7686 7702 7715 7718 7720 7726 7736 7741 7752 7754 7760 7766 7768
 7784 7797 7804 7807 7808 7810 7825 7830 7836 7848 7867 7895 7909 7912
 7929 7930 7946 7957 7971 7981 7985 7988 7990 8044 8045 8081 8086 8114
 8115 8131 8134 8136 8142 8160 8163 8171 8173 8177 8187 8219 8229 8238
 8241 8255 8274 8277 8299 8309 8327 8334 8339 8346 8347 8351 8360 8368
 8389 8390 8399 8432 8434 8436 8450 8475 8478 8482 8502 8505 8508 8514
 8524 8525 8534 8541 8599 8615 8619 8620 8638 8643 8678 8692 8704 8709
 8720 8725 8728 8736 8741 8746 8749 8768 8778 8788 8790 8800 8804 8806
 8807 8840 8844 8847 8885 8892 8914 8923 8925 8928 8935 8941 8944 8952
 8959 8961 8967 8969 8975 8980 8997 9058 9074 9086 9096 9098 9143 9152
 9155 9156 9193 9197 9216 9226 9243 9263 9270 9281 9288 9289 9295 9306
 9315 9324 9328 9334 9346 9351 9369 9388 9404 9422 9432 9433 9434 9443
 9453 9458 9459 9468 9471 9472 9492 9493 9503 9512 9525 9529 9538 9540
```

```
 9547 9548 9549]
[   0    1    2 ... 9554 9555 9556] [  11   29   44   55   67   69   74   86   90  102  118  122
134  140
  141  142  160  168  170  178  184  188  190  204  205  221  225  231
  238  246  271  291  317  321  356  377  383  386  387  390  395  403
  416  418  419  436  452  453  496  497  503  513  515  556  558  581
  583  594  597  604  618  633  643  653  667  674  695  696  699  707
  708  716  723  731  749  757  777  820  821  852  885  894  897  898
  913  927  940  955  958  960  967  969  976  999 1006 1016 1020 1024
 1027 1042 1052 1069 1112 1120 1121 1133 1134 1139 1153 1157 1177 1179
 1193 1199 1200 1213 1216 1217 1251 1258 1267 1278 1280 1310 1314 1353
 1360 1366 1376 1390 1398 1414 1421 1422 1427 1438 1443 1446 1448 1449
 1466 1475 1491 1507 1519 1530 1535 1548 1567 1580 1586 1593 1595 1605
 1633 1648 1656 1658 1660 1661 1664 1673 1674 1685 1701 1715 1745 1756
 1759 1767 1780 1787 1792 1795 1796 1807 1810 1816 1829 1845 1848 1865
 1873 1876 1884 1888 1890 1902 1903 1928 1932 1939 1942 1943 1947 1955
 1960 1963 1993 1996 2013 2023 2029 2032 2034 2039 2056 2087 2092 2104
 2106 2108 2109 2117 2129 2137 2141 2152 2158 2179 2185 2217 2223 2232
 2252 2268 2272 2289 2290 2307 2313 2315 2323 2325 2327 2337 2348 2380
 2381 2393 2407 2420 2421 2431 2432 2441 2445 2452 2454 2457 2467 2469
 2470 2478 2479 2481 2485 2506 2521 2525 2538 2559 2576 2578 2580 2581
 2585 2586 2588 2602 2604 2630 2631 2686 2692 2696 2697 2717 2728 2735
 2736 2742 2749 2757 2760 2764 2765 2767 2791 2819 2833 2846 2848 2857
 2862 2866 2883 2886 2901 2922 2931 2941 3011 3018 3033 3041 3062 3069
 3070 3075 3090 3097 3102 3126 3130 3133 3150 3159 3177 3185 3190 3198
 3207 3210 3211 3224 3227 3235 3260 3274 3293 3304 3314 3320 3323 3355
 3360 3378 3386 3389 3399 3409 3418 3425 3437 3443 3448 3459 3460 3497
 3502 3505 3526 3547 3559 3565 3570 3595 3596 3604 3605 3608 3615 3633
 3642 3663 3668 3678 3680 3690 3702 3703 3712 3722 3723 3738 3757 3785
 3789 3794 3804 3819 3824 3827 3842 3851 3855 3856 3860 3863 3868 3869
 3878 3881 3890 3893 3895 3897 3898 3902 3907 3916 3924 3927 3942 3958
 3964 3982 4003 4014 4023 4029 4031 4038 4039 4041 4045 4071 4079 4086
 4092 4095 4097 4105 4131 4145 4158 4165 4169 4191 4207 4213 4216 4225
 4227 4237 4266 4286 4291 4297 4305 4348 4350 4353 4363 4364 4372 4378
 4383 4385 4393 4397 4404 4438 4443 4462 4463 4478 4482 4522 4523 4539
 4546 4552 4564 4566 4597 4615 4633 4637 4682 4695 4698 4699 4718 4731
 4735 4749 4766 4773 4774 4779 4781 4786 4792 4810 4813 4818 4822 4833
 4835 4862 4863 4867 4868 4873 4894 4899 4903 4910 4911 4916 4924 4927
 4963 4970 4979 5002 5007 5012 5026 5031 5032 5043 5051 5056 5063 5068
 5069 5072 5076 5077 5085 5087 5090 5101 5106 5110 5122 5141 5150 5164
 5168 5179 5182 5185 5193 5199 5206 5207 5219 5225 5230 5263 5268 5274
 5277 5283 5287 5291 5292 5329 5330 5331 5335 5372 5387 5403 5414 5445
 5505 5521 5526 5540 5543 5551 5553 5573 5574 5594 5598 5603 5635 5639
 5675 5679 5705 5714 5734 5743 5751 5764 5771 5776 5785 5788 5808 5811
 5816 5826 5834 5847 5853 5858 5865 5879 5880 5894 5906 5926 5928 5933
 5942 5945 5954 5978 5993 6007 6015 6019 6022 6030 6039 6066 6076 6079
 6082 6086 6093 6109 6124 6126 6127 6134 6139 6146 6149 6156 6173 6177
 6182 6187 6218 6233 6237 6252 6260 6278 6283 6289 6293 6299 6302 6303
 6336 6348 6361 6365 6367 6383 6393 6401 6411 6416 6421 6425 6432 6434
 6444 6446 6457 6530 6556 6570 6572 6574 6578 6596 6605 6614 6616 6635
 6638 6671 6672 6686 6688 6693 6698 6702 6719 6737 6744 6751 6755 6768
 6770 6772 6773 6794 6799 6800 6802 6825 6841 6865 6886 6890 6898 6908
 6911 6914 6921 6943 6960 6967 6974 6976 6996 7001 7020 7022 7023 7034
 7043 7050 7062 7069 7070 7100 7106 7144 7150 7152 7154 7159 7160 7169
 7173 7187 7193 7259 7289 7315 7324 7326 7356 7363 7376 7380 7391 7396
 7397 7416 7426 7432 7451 7452 7461 7465 7468 7470 7511 7532 7533 7540
 7542 7579 7600 7616 7626 7637 7642 7651 7658 7663 7668 7713 7724 7734
 7746 7749 7758 7770 7771 7777 7780 7781 7802 7803 7821 7834 7851 7853
 7862 7863 7866 7879 7889 7893 7896 7900 7903 7905 7906 7910 7911 7913
 7914 7916 7920 7933 7935 7970 7984 8006 8008 8027 8029 8049 8060 8102
 8123 8139 8175 8176 8178 8182 8210 8246 8247 8249 8254 8257 8261 8279
 8289 8291 8307 8319 8325 8332 8348 8353 8364 8373 8396 8414 8419 8462
 8490 8496 8500 8504 8526 8527 8550 8555 8564 8565 8566 8623 8633 8661
 8699 8707 8708 8716 8717 8719 8735 8756 8763 8773 8783 8794 8802 8808
 8816 8818 8821 8823 8827 8834 8837 8846 8850 8860 8869 8871 8883 8900
 8931 8934 8940 8949 8953 8970 8982 9005 9011 9024 9057 9073 9089 9093
 9108 9114 9139 9142 9144 9145 9151 9157 9158 9180 9185 9206 9208 9217
 9220 9224 9227 9231 9232 9238 9240 9254 9271 9274 9275 9282 9285 9317
 9327 9336 9337 9339 9374 9377 9380 9383 9384 9392 9396 9398 9403 9411
 9431 9461 9483 9484 9486 9489 9490 9496 9499 9500 9510 9513 9521 9533
```

```
 9543 9550 9551]
[   0    1    2 ... 9554 9555 9556] [  17   25   31   41   45   57   66   71   77   83   92  111
113  139
  163  176  213  215  229  236  244  251  256  261  279  282  286  312
  325  326  336  342  360  370  381  412  425  464  476  479  482  510
  519  520  541  551  561  563  575  579  582  595  599  614  623  639
  646  649  677  684  685  688  702  704  713  722  724  728  743  756
  762  767  776  781  800  816  817  829  834  836  838  853  855  874
  876  884  893  901  903  906  934  943  959  961  974  992  994  995
 1002 1023 1026 1033 1034 1043 1086 1099 1100 1103 1104 1108 1115 1123
 1136 1145 1165 1187 1236 1240 1248 1249 1253 1286 1293 1299 1313 1320
 1327 1331 1332 1337 1341 1342 1343 1346 1371 1372 1382 1383 1384 1388
 1452 1453 1456 1480 1489 1492 1493 1506 1513 1524 1546 1553 1581 1602
 1610 1620 1622 1623 1625 1628 1657 1689 1710 1739 1741 1760 1769 1778
 1786 1800 1804 1820 1827 1833 1840 1855 1859 1863 1905 1918 1926 1964
 1971 1975 1986 1992 2018 2019 2028 2035 2040 2052 2066 2067 2072 2107
 2132 2133 2139 2146 2148 2155 2171 2182 2183 2191 2203 2210 2219 2222
 2228 2237 2246 2249 2259 2264 2273 2294 2297 2301 2331 2335 2352 2353
 2355 2358 2361 2376 2384 2402 2404 2416 2419 2442 2447 2465 2475 2476
 2477 2480 2483 2501 2528 2539 2561 2562 2573 2579 2583 2584 2592 2612
 2632 2637 2643 2652 2662 2664 2673 2683 2700 2714 2720 2744 2754 2756
 2761 2789 2790 2793 2795 2803 2807 2817 2818 2836 2840 2879 2887 2893
 2898 2926 2939 2942 2951 2960 2967 2977 2982 2985 2992 2999 3003 3014
 3032 3035 3037 3040 3072 3073 3074 3078 3092 3096 3104 3106 3107 3123
 3132 3147 3152 3186 3187 3204 3231 3239 3242 3243 3253 3262 3264 3267
 3273 3283 3287 3288 3329 3335 3346 3348 3353 3354 3357 3361 3366 3402
 3453 3463 3468 3486 3488 3495 3498 3510 3511 3513 3521 3522 3525 3528
 3534 3542 3543 3568 3575 3592 3593 3601 3609 3611 3616 3621 3627 3629
 3646 3662 3685 3687 3691 3694 3714 3715 3748 3750 3754 3760 3771 3810
 3813 3814 3816 3848 3850 3865 3871 3874 3875 3896 3900 3911 3913 3922
 3936 3938 3943 3947 3948 3988 3994 4000 4001 4005 4013 4019 4030 4032
 4040 4051 4064 4069 4080 4094 4103 4108 4134 4138 4159 4160 4163 4164
 4190 4192 4197 4203 4222 4223 4232 4235 4241 4271 4278 4280 4282 4285
 4314 4330 4335 4337 4349 4368 4389 4405 4409 4428 4447 4460 4476 4479
 4506 4507 4512 4513 4514 4520 4535 4540 4574 4586 4589 4591 4599 4608
 4619 4627 4628 4631 4634 4638 4659 4664 4666 4667 4677 4696 4726 4736
 4746 4752 4760 4777 4778 4788 4815 4829 4837 4855 4857 4864 4877 4882
 4907 4925 4940 4947 4955 4960 4961 4965 4967 4973 4978 4990 4995 5024
 5027 5034 5035 5045 5049 5065 5079 5081 5092 5094 5099 5100 5135 5139
 5147 5159 5161 5165 5166 5178 5180 5189 5201 5213 5228 5233 5240 5255
 5270 5271 5273 5300 5302 5317 5339 5341 5346 5348 5364 5367 5368 5369
 5370 5376 5378 5392 5393 5436 5454 5459 5462 5470 5493 5497 5530 5547
 5559 5562 5564 5571 5591 5596 5620 5624 5633 5634 5638 5670 5671 5678
 5710 5711 5737 5745 5767 5796 5805 5813 5819 5828 5836 5844 5849 5852
 5867 5868 5869 5878 5902 5903 5932 5935 5944 5947 5972 5986 5987 5988
 5990 5995 5996 6021 6028 6034 6043 6046 6048 6057 6094 6098 6107 6116
 6118 6132 6135 6145 6152 6164 6189 6198 6201 6207 6212 6240 6247 6256
 6266 6269 6275 6279 6311 6330 6337 6343 6366 6381 6382 6399 6407 6420
 6427 6433 6439 6441 6452 6455 6459 6461 6469 6476 6500 6516 6523 6537
 6544 6549 6550 6557 6564 6576 6580 6587 6608 6609 6610 6627 6632 6634
 6656 6659 6660 6666 6668 6669 6673 6678 6734 6782 6795 6796 6806 6809
 6816 6823 6828 6836 6842 6844 6847 6868 6869 6872 6882 6893 6907 6910
 6935 6949 6957 6964 6975 6977 6987 7004 7030 7040 7064 7067 7089 7090
 7095 7096 7097 7117 7125 7147 7167 7179 7186 7188 7215 7220 7225 7237
 7245 7261 7265 7276 7280 7285 7296 7299 7313 7346 7349 7364 7381 7387
 7401 7427 7448 7449 7450 7482 7500 7504 7508 7521 7524 7527 7560 7571
 7572 7573 7580 7584 7589 7595 7604 7618 7630 7648 7649 7650 7656 7659
 7665 7676 7677 7679 7688 7694 7706 7719 7721 7723 7775 7776 7809 7811
 7837 7842 7857 7858 7871 7884 7894 7897 7908 7917 7931 7944 7952 7967
 7973 7978 7979 7983 7997 8034 8037 8047 8050 8051 8058 8068 8069 8071
 8077 8080 8084 8090 8096 8106 8108 8110 8124 8141 8152 8153 8164 8172
 8186 8189 8190 8191 8201 8207 8215 8220 8222 8224 8267 8278 8287 8294
 8295 8302 8303 8321 8326 8328 8344 8358 8378 8391 8403 8411 8458 8461
 8469 8473 8479 8481 8494 8495 8499 8509 8631 8644 8646 8659 8666 8671
 8683 8693 8697 8714 8718 8737 8753 8757 8761 8771 8777 8810 8811 8814
 8833 8839 8853 8863 8876 8886 8893 8894 8927 8955 8956 8962 8971 8973
 8988 9003 9006 9016 9018 9019 9031 9034 9040 9041 9047 9050 9060 9087
 9090 9113 9119 9124 9135 9160 9165 9181 9190 9198 9211 9213 9215 9236
 9239 9246 9247 9248 9257 9265 9269 9272 9273 9300 9318 9326 9333 9341
```

```
 9348 9353 9360 9381 9401 9410 9426 9430 9438 9440 9475 9482 9509 9511
 9530 9536 9545]
```

In [181]:
```
# Import the cross-validation library
from sklearn.model_selection import cross_val_score

# Define a new random forest classifier
rf_class = RandomForestClassifier(n_estimators = 10)
```

In [182]:
```
# Use cross validation with 10 interations???? This is where the random forest classifier comes int
o play
cvResult = cross_val_score(rf_class, dataInput, dataOutput, scoring = "accuracy", cv = 10)
print(cvResult)
```

```
[0.61297071 0.63702929 0.66527197 0.60564854 0.65690377 0.65585774
 0.57845188 0.48062827 0.55602094 0.59162304]
```

In [183]:
```
# Check how accurate the cross-validation result is in terms of a percentage.
accuracy = cvResult.mean() * 100
print("Accuracy of the random forest classifier is ", accuracy)
```

```
Accuracy of the random forest classifier is  60.4040614252229
```

In [ ]:
```

```