In [1]:
```
import pandas as pd
import csv
from scipy.stats import trim_mean
from scipy.stats.mstats import mode, gmean, hmean
from scipy.stats import skew
from scipy.stats import kurtosis
from sklearn.model_selection import train_test_split
from sklearn import linear_model
from scipy.stats import zscore
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from scipy.stats import pearsonr
from sklearn.metrics import mean_absolute_error
import matplotlib
import numpy as np
import scipy.stats
%matplotlib inline

data = pd.read_csv(r"D:\\reachout analytics\\case studies\\ipynb files
\\Bank_ Data.csv")
data.head(10)
```

Out[1]:

	Obs	account_id	cardwdln	cardwdlt	cashcrn	cashcrt	cashwdn	cashwdt	bankcolt	ban
0	1	2	0	0	12	48400.0	237	1001191.0	1537936	
1	2	19	0	0	17	45800.0	143	762135.7	741807	
2	3	25	0	0	54	1488172.0	99	1215437.6	0	
3	4	37	0	0	19	494494.0	51	375845.2	0	
4	5	38	0	0	14	49700.0	37	156679.0	256060	
5	6	67	0	0	2	6800.0	164	1758719.0	2261205	
6	7	97	5	13200	35	635480.0	81	365494.2	0	
7	8	103	0	0	38	867846.0	90	757291.9	0	

	Obs	account_id	cardwdln	cardwdlt	cashcrn	cashcrt	cashwdn	cashwdt	bankcolt	ban
8	9	105	0	0	14	252833.0	28	230921.2	0	
9	10	110	5	14800	32	745897.0	79	515150.4	0	

10 rows × 40 columns

In [2]:
```
### SORTING THE DATA ###
sorted_data = data.sort_values
sorted_data
```

Out[2]:
```
<bound method DataFrame.sort_values of      Obs  account_id  cardwdln  
cardwdlt  cashcrn    cashcrt  cashwdn  \
0     1          2         0        0       12    48400.0      237   

1     2         19         0        0       17    45800.0      143   

2     3         25         0        0       54  1488172.0       99   

3     4         37         0        0       19   494494.0       51   

4     5         38         0        0       14    49700.0       37   

5     6         67         0        0        2     6800.0      164   

6     7         97         5    13200       35   635480.0       81   

7     8        103         0        0       38   867846.0       90   

8     9        105         0        0       14   252833.0       28   

9    10        110         5    14800       32   745897.0       79   

10   11        132         2     7500       47  1278771.0      105   

11   12        173         0        0       95  1314120.0      176   
```

12	13	176	0	0	53	1475516.0	102
13	14	226	6	14500	33	484714.0	44
14	15	276	1	3600	1	700.0	28
15	16	290	0	0	100	2092453.0	171
16	17	303	2	5900	33	504360.0	80
17	18	309	0	0	22	426020.0	65
18	19	314	0	0	4	8300.0	98
19	20	319	0	0	74	1713632.0	138
20	21	330	0	0	26	616210.0	73
21	22	339	2	6000	2	4000.0	98
22	23	344	0	0	67	939342.0	166
23	24	347	0	0	89	1990266.0	207
24	25	349	0	0	32	177709.0	38
25	26	378	0	0	73	1974944.0	114
26	27	426	0	0	89	1560877.0	129
27	28	440	0	0	28	435438.0	67
28	29	442	0	0	18	427443.0	24
29	30	472	0	0	2	2400.0	86
..	...	...	...	...	...	...	...
652	653	10915	0	0	6	27100.0	49

653	654	10940	0	0	66	1657264.0	205
654	655	10942	0	0	119	3150073.0	179
655	656	10954	16	35000	31	410777.0	69
656	657	10963	0	0	70	1684174.0	146
657	658	10973	0	0	118	3137980.0	239
658	659	11013	11	33100	4	3300.0	225
659	660	11021	0	0	106	2548952.0	241
660	661	11027	0	0	14	69400.0	71
661	662	11042	11	31300	2	4000.0	135
662	663	11054	0	0	31	194900.0	72
663	664	11065	0	0	71	896219.0	214
664	665	11069	0	0	52	1192751.0	139
665	666	11079	4	8000	9	20100.0	145
666	667	11096	0	0	28	636472.0	58
667	668	11111	0	0	121	2816451.0	202
668	669	11135	0	0	70	1066663.0	191
669	670	11138	8	17900	90	2702121.0	162
670	671	11141	5	13200	45	889272.0	68
671	672	11186	2	3700	8	41700.0	159

```
672  673        11231         0         0         2     6800.0        76

673  674        11244         0         0        44  1012868.0        72

674  675        11265         0         0        97   650086.0       170

675  676        11271         0         0        17    52300.0        80

676  677        11317         0         0        36  1124913.0        70

677  678        11327         0         0        16   315374.0        33

678  679        11328         0         0        38   690261.0        78

679  680        11349         0         0         5    14800.0       136

680  681        11359        22     57400         2     4000.0       148

681  682        11362         0         0        43   680055.0       103


       cashwdt  bankcolt  bankcoln  ...  bankrtd   othcrnd  othcrtd  ac
ardwdl  \
0    1001191.0   1537936        70  ...  259.248  0.032802   5.0223
  0.00
1     762135.7    741807        45  ...   14.799  0.065982   4.0965
  0.00
2    1215437.6         0         0  ...  280.740  0.031603   6.9979
  0.00
3     375845.2         0         0  ...  150.559  0.034000   5.0704
  0.00
4     156679.0    256060        17  ...  209.815  0.033333   4.9124
  0.00
5    1758719.0   2261205        50  ...  313.239  0.032595   7.1265
  0.00
6     365494.2         0         0  ...  216.126  0.032990   5.6085   2
640.00
7     757291.9         0         0  ...   79.018  0.044834   8.3248
  0.00
```

```
8      230921.2        0          0  ...   13.633  0.029685   4.9308
  0.00
9      515150.4        0          0  ...  164.780  0.032330   6.4017   2
960.00
10    1154757.0        0          0  ...   91.745  0.033195   6.7269   3
750.00
11     663257.6        0          0  ...  319.970  0.032778   4.6250
  0.00
12    1259867.0        0          0  ...  171.735  0.033295   6.6069
  0.00
13     214448.2        0          0  ...  318.604  0.032544   6.0380   2
416.67
14     283509.8   388932         13  ...  118.986  0.033505   7.7843   3
600.00
15    1611962.0        0          0  ...  245.611  0.032345   6.6470
  0.00
16     344350.4        0          0  ...  131.607  0.032621   6.0165   2
950.00
17     402199.0        0          0  ...    3.091  0.032520   3.7283
  0.00
18     909638.0  1402089         35  ...  448.711  0.032710   4.4255
  0.00
19    1401114.0        0          0  ...  207.323  0.032451   7.7115
  0.00
20     552032.0        0          0  ...   54.039  0.032595   5.2111
  0.00
21    1147291.0  1427611         28  ...  224.399  0.033294   9.9705   3
000.00
22     541517.6        0          0  ...  189.066  0.032866   5.2193
  0.00
23    1773345.8        0          0  ...   93.073  0.033080   5.6760
  0.00
24      82944.4        0          0  ...  139.250  0.033275   2.5040
  0.00
25    1614008.8        0          0  ...  369.909  0.033632   8.9553
  0.00
26     853726.3        0          0  ...  368.935  0.032831   3.9606
  0.00
27     252377.4        0          0  ...  171.149  0.032476   4.5966
```

```
  0.00
28    269202.8         0         0  ...  165.215  0.026521   3.1814
  0.00
29   1224322.0   1402231        26  ...  192.113  0.032059   7.6954
  0.00
..         ...       ...       ...  ...      ...       ...      ...
   ...
652   166004.0    365882        22  ...  303.866  0.032164   2.9961
  0.00
653   978026.8         0         0  ...  327.511  0.032558   6.6163
  0.00
654  2390833.0         0         0  ...  432.812  0.032934   7.1699
  0.00
655   330850.4         0         0  ...    7.403  0.032768   5.4031   2
187.50
656  1537978.6         0         0  ...   58.441  0.033308   7.6207
  0.00
657  2940206.0         0         0  ...   74.196  0.033157   6.8546
  0.00
658  2161662.0   3210544        70  ...  443.063  0.032619   7.8582   3
009.09
659  2311770.8         0         0  ...   78.667  0.066023  13.2786
  0.00
660   283005.0    378790        29  ...  134.273  0.032658   4.3019
  0.00
661  1202778.2   1845737        47  ...  404.953  0.032821   6.9895   2
845.45
662   286117.0    565896        35  ...  400.547  0.032895   4.2798
  0.00
663   797634.4         0         0  ...   33.509  0.032655   4.3646
  0.00
664  1029923.0         0         0  ...   41.746  0.032875   6.8706
  0.00
665  1863768.8   2039687        38  ...   85.700  0.033130  10.4943   2
000.00
666   374237.4         0         0  ...  267.330  0.031957   4.6907
  0.00
667  2305464.0         0         0  ...  202.829  0.033155   6.9177
  0.00
```

```
668   654712.2         0         0  ...  193.226  0.032495   5.4656
  0.00
669  2550009.0         0         0  ...   73.008  0.033306   9.4908   2
237.50
670   743488.0         0         0  ...   65.315  0.031977   7.7208   2
640.00
671  1447077.4   2099710        49  ...  412.592  0.032710   8.1499   1
850.00
672   788007.8   1116073        23  ...  388.661  0.033141   7.5961
  0.00
673   694144.0         0         0  ...  422.212  0.032033   4.8386
  0.00
674   367403.6         0         0  ...  120.055  0.032614   2.7872
  0.00
675   165775.4    476935        39  ...  289.730  0.032554   2.7717
  0.00
676   934000.0         0         0  ...  194.769  0.033457   8.1396
  0.00
677   203131.4         0         0  ...   86.274  0.031674   7.3376
  0.00
678   608247.0         0         0  ...   29.733  0.031807   6.2875
  0.00
679  1271219.0   1973436        43  ...  525.078  0.032700   5.6572
  0.00
680  1105500.6   1499277        51  ...  176.250  0.032861   5.3893   2
609.09
681   336301.8         0         0  ...  263.576  0.032368   3.5909
  0.00

      acashcr   acashwd  abankcol   abankr   aothcr good
0     4033.33   4224.43  21970.51  6216.12  153.107  Yes
1     2694.12   5329.62  16484.60  2523.20   62.084   No
2    27558.74  12277.15      0.00  2674.57  221.432  Yes
3    26026.00   7369.51      0.00  1750.69  149.129   No
4     3550.00   4234.57  15062.35  2377.90  147.371  Yes
5     3400.00  10723.90  45224.10  4142.31  218.640  Yes
6    18156.57   4512.27      0.00  1732.58  170.006  Yes
7    22838.05   8414.35      0.00  7370.20  185.678   No
8    18059.50   8247.19      0.00  7348.00  166.106  Yes
```

9	23309.28	6520.89	0.00	2309.50	198.010	Yes
10	27207.89	10997.69	0.00	7370.20	202.647	Yes
11	13832.84	3768.51	0.00	2481.10	141.100	Yes
12	27839.92	12351.64	0.00	1662.02	198.434	Yes
13	14688.30	4873.82	0.00	8974.00	185.532	Yes
14	700.00	10125.35	29917.85	5129.61	232.331	Yes
15	20924.53	9426.68	0.00	2847.55	205.502	Yes
16	15283.64	4304.38	0.00	3656.22	184.438	Yes
17	19364.55	6187.68	0.00	1901.00	114.645	Yes
18	2075.00	9282.02	40059.69	9414.14	135.294	Yes
19	23157.19	10153.00	0.00	1926.79	237.637	Yes
20	23700.38	7562.08	0.00	6908.00	159.876	Yes
21	2000.00	11707.05	50986.11	3253.78	299.471	Yes
22	14020.03	3262.15	0.00	3341.97	158.803	Yes
23	22362.54	8566.89	0.00	7801.20	171.582	No
24	5553.41	2182.75	0.00	4417.33	75.253	Yes
25	27054.03	14157.97	0.00	2598.10	266.270	Yes
26	17537.94	6618.03	0.00	4154.43	120.636	No
27	15551.36	3766.83	0.00	4517.11	141.538	Yes
28	23746.83	11216.78	0.00	5043.00	119.959	No
29	1200.00	14236.30	53931.96	8200.20	240.038	No
..	...	...	...	...	...	...
652	4516.67	3387.84	16631.00	2734.80	93.150	Yes
653	25110.06	4770.86	0.00	6741.85	203.216	Yes
654	26471.20	13356.61	0.00	8213.59	217.704	Yes
655	13250.87	4794.93	0.00	312.00	164.886	Yes
656	24059.63	10534.10	0.00	3216.70	228.793	Yes
657	26593.05	12302.12	0.00	3216.70	206.730	Yes
658	825.00	9607.39	45864.91	4978.08	240.911	Yes
659	24046.72	9592.41	0.00	7040.70	201.119	No
660	4957.14	3985.99	13061.72	3137.76	131.728	Yes
661	2000.00	8909.47	39271.00	7340.42	212.955	Yes
662	6287.10	3973.85	16168.46	7223.42	130.106	Yes
663	12622.80	3727.26	0.00	1445.00	133.659	Yes
664	22937.52	7409.52	0.00	2482.00	208.995	Yes
665	2233.33	12853.58	53675.97	8191.50	316.763	Yes
666	22731.14	6452.37	0.00	5147.82	146.779	Yes
667	23276.45	11413.19	0.00	4118.83	208.651	Yes
668	15238.04	3427.81	0.00	3964.26	168.200	Yes

```
669  30023.57  15740.80      0.00  3744.70  284.956  Yes
670  19761.60  10933.65      0.00  3744.70  241.450  Yes
671   5212.50   9101.12  42851.22  8584.20  249.155  Yes
672   3400.00  10368.52  48524.91  5504.71  229.204  Yes
673  23019.73   9640.89      0.00  9778.98  151.048  Yes
674   6701.92   2161.20      0.00  1790.13   85.462  Yes
675   3076.47   2072.19  12229.10  2711.69   85.141  Yes
676  31247.58  13342.86      0.00  3380.19  243.283  Yes
677  19710.88   6155.50      0.00  3177.75  231.657  Yes
678  18164.76   7798.04      0.00  4674.00  197.680  Yes
679   2960.00   9347.20  45893.86  8967.25  173.005  Yes
680   2000.00   7469.60  29397.59  2630.19  164.004  Yes
681  15815.23   3265.07      0.00  1933.99  110.939  Yes

[682 rows x 40 columns]>
```

In [3]:
```
data.shape
```

Out[3]:
```
(682, 40)
```

In [4]:
```
### DROPPING Obs and account_id AS THEY ARE NOT NEEDED ###
data.drop(['Obs','account_id'],axis=1, inplace=True)
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 682 entries, 0 to 681
Data columns (total 38 columns):
cardwdln     682 non-null int64
cardwdlt     682 non-null int64
cashcrn      682 non-null int64
cashcrt      682 non-null float64
cashwdn      682 non-null int64
cashwdt      682 non-null float64
bankcolt     682 non-null int64
bankcoln     682 non-null int64
bankrn       682 non-null int64
bankrt       682 non-null float64
othcrn       682 non-null int64
othcrt       682 non-null float64
```

```
days         682 non-null int64
sex          682 non-null object
card         682 non-null object
age          682 non-null float64
second       682 non-null object
frequency    682 non-null object
region       682 non-null object
cardwdlnd    682 non-null float64
cardwdltd    682 non-null float64
cashcrnd     682 non-null float64
cashcrtd     682 non-null float64
cashwdnd     682 non-null float64
cashwdtd     682 non-null float64
bankcoltd    682 non-null float64
bankcolnd    682 non-null float64
bankrnd      682 non-null float64
bankrtd      682 non-null float64
othcrnd      682 non-null float64
othcrtd      682 non-null float64
acardwdl     682 non-null float64
acashcr      682 non-null float64
acashwd      682 non-null float64
abankcol     682 non-null float64
abankr       682 non-null float64
aothcr       682 non-null float64
good         682 non-null object
dtypes: float64(23), int64(9), object(6)
memory usage: 202.5+ KB
None
```

In [5]:
```
data.shape
```

Out[5]:
```
(682, 38)
```

In [6]:
```
data.describe()
```

Out[6]:

	cardwdln	cardwdlt	cashcrn	cashcrt	cashwdn	cashwdt	bankc

	cardwdln	cardwdlt	cashcrn	cashcrt	cashwdn	cashwdt	bankc
count	682.000000	682.000000	682.000000	6.820000e+02	682.000000	6.820000e+02	6.820000e+
mean	1.414956	3640.175953	42.523460	7.964454e+05	116.917889	8.883424e+05	3.833494e+
std	4.098560	10510.752426	33.808959	7.961603e+05	61.467198	6.698405e+05	7.106200e+
min	0.000000	0.000000	1.000000	2.000000e+02	13.000000	3.050220e+04	0.000000e+
25%	0.000000	0.000000	10.000000	4.080000e+04	70.000000	3.545102e+05	0.000000e+
50%	0.000000	0.000000	39.000000	6.276590e+05	99.000000	7.008136e+05	0.000000e+
75%	0.000000	0.000000	63.000000	1.235188e+06	161.000000	1.206393e+06	5.276402e+
max	34.000000	83000.000000	148.000000	3.708832e+06	324.000000	3.392850e+06	3.552197e+

8 rows × 32 columns

In [7]:

```
median = data.median()
print(median)
```

```
cardwdln            0.000000
cardwdlt            0.000000
cashcrn            39.000000
cashcrt        627659.000000
cashwdn            99.000000
cashwdt        700813.600000
bankcolt            0.000000
bankcoln            0.000000
bankrn             44.500000
bankrt         196321.750000
othcrn             38.000000
othcrt           6489.400000
days             1070.000000
age                41.374400
cardwdlnd           0.000000
cardwdltd           0.000000
cashcrnd            0.037707
cashcrtd          586.135000
```

```
cashwdnd          0.092975
cashwdtd        654.120000
bankcoltd         0.000000
bankcolnd         0.000000
bankrnd           0.043400
bankrtd         182.086000
othcrnd           0.032757
othcrtd           6.088000
acardwdl          0.000000
acashcr       15970.000000
acashwd        7518.560000
abankcol          0.000000
abankr         4211.305000
aothcr          172.742500
dtype: float64
```

In [8]:

```
mode = data.mode()
print(mode)
##### In rest of the column all element are mode because they have the
 same frequency of occurrence.######
```

```
   cardwdln  cardwdlt  cashcrn  cashcrt  cashwdn    cashwdt  bankcolt
  \
0       0.0       0.0      1.0    400.0     51.0    30502.2       0.0

1       NaN       NaN      NaN    700.0     70.0    41031.4       NaN

2       NaN       NaN      NaN      NaN      NaN    53604.4       NaN

3       NaN       NaN      NaN      NaN      NaN    61319.0       NaN

4       NaN       NaN      NaN      NaN      NaN    61660.6       NaN

5       NaN       NaN      NaN      NaN      NaN    63209.8       NaN

6       NaN       NaN      NaN      NaN      NaN    71059.0       NaN

7       NaN       NaN      NaN      NaN      NaN    71193.2       NaN
```

8	NaN	NaN	NaN	NaN	NaN	75879.6	NaN
9	NaN	NaN	NaN	NaN	NaN	82944.4	NaN
10	NaN	NaN	NaN	NaN	NaN	83946.0	NaN
11	NaN	NaN	NaN	NaN	NaN	84669.8	NaN
12	NaN	NaN	NaN	NaN	NaN	86975.8	NaN
13	NaN	NaN	NaN	NaN	NaN	91409.8	NaN
14	NaN	NaN	NaN	NaN	NaN	91797.4	NaN
15	NaN	NaN	NaN	NaN	NaN	92950.4	NaN
16	NaN	NaN	NaN	NaN	NaN	93061.8	NaN
17	NaN	NaN	NaN	NaN	NaN	99299.0	NaN
18	NaN	NaN	NaN	NaN	NaN	99752.0	NaN
19	NaN	NaN	NaN	NaN	NaN	102039.8	NaN
20	NaN	NaN	NaN	NaN	NaN	103749.4	NaN
21	NaN	NaN	NaN	NaN	NaN	108023.4	NaN
22	NaN	NaN	NaN	NaN	NaN	108864.4	NaN
23	NaN	NaN	NaN	NaN	NaN	109002.2	NaN
24	NaN	NaN	NaN	NaN	NaN	113677.4	NaN
25	NaN	NaN	NaN	NaN	NaN	114954.8	NaN
26	NaN	NaN	NaN	NaN	NaN	122025.6	NaN
27	NaN	NaN	NaN	NaN	NaN	124793.2	NaN

28	NaN	NaN	NaN	NaN	NaN	125960.6	NaN
29	NaN	NaN	NaN	NaN	NaN	126009.4	NaN
..	...	...	...	...	...	...	...
652	NaN	NaN	NaN	NaN	NaN	2330311.0	NaN
653	NaN	NaN	NaN	NaN	NaN	2353230.6	NaN
654	NaN	NaN	NaN	NaN	NaN	2375511.0	NaN
655	NaN	NaN	NaN	NaN	NaN	2377576.4	NaN
656	NaN	NaN	NaN	NaN	NaN	2385261.6	NaN
657	NaN	NaN	NaN	NaN	NaN	2390833.0	NaN
658	NaN	NaN	NaN	NaN	NaN	2409950.4	NaN
659	NaN	NaN	NaN	NaN	NaN	2455890.6	NaN
660	NaN	NaN	NaN	NaN	NaN	2464280.0	NaN
661	NaN	NaN	NaN	NaN	NaN	2487744.6	NaN
662	NaN	NaN	NaN	NaN	NaN	2523352.0	NaN
663	NaN	NaN	NaN	NaN	NaN	2549369.2	NaN
664	NaN	NaN	NaN	NaN	NaN	2550009.0	NaN
665	NaN	NaN	NaN	NaN	NaN	2591834.4	NaN
666	NaN	NaN	NaN	NaN	NaN	2603568.4	NaN
667	NaN	NaN	NaN	NaN	NaN	2635616.0	NaN

```
668       NaN       NaN       NaN       NaN       NaN  2643253.0       NaN

669       NaN       NaN       NaN       NaN       NaN  2658630.4       NaN

670       NaN       NaN       NaN       NaN       NaN  2725216.0       NaN

671       NaN       NaN       NaN       NaN       NaN  2766369.4       NaN

672       NaN       NaN       NaN       NaN       NaN  2793697.6       NaN

673       NaN       NaN       NaN       NaN       NaN  2794748.4       NaN

674       NaN       NaN       NaN       NaN       NaN  2877618.2       NaN

675       NaN       NaN       NaN       NaN       NaN  2940206.0       NaN

676       NaN       NaN       NaN       NaN       NaN  3022515.8       NaN

677       NaN       NaN       NaN       NaN       NaN  3044910.2       NaN

678       NaN       NaN       NaN       NaN       NaN  3051500.0       NaN

679       NaN       NaN       NaN       NaN       NaN  3341448.2       NaN

680       NaN       NaN       NaN       NaN       NaN  3366800.0       NaN

681       NaN       NaN       NaN       NaN       NaN  3392850.2       NaN


     bankcoln  bankrn    bankrt  ...  bankrtd   othcrnd  othcrtd acardw
dl  \
0         0.0    12.0    5580.0  ...   11.071  0.033333   4.4255
0.0
1         NaN     NaN   20185.6  ...   14.799       NaN   5.4402      N
aN
2         NaN     NaN   20517.0  ...   33.470       NaN   7.1265      N
aN
3         NaN     NaN   37163.2  ...   50.839       NaN      NaN      N
aN
```

```
4         NaN      NaN   67470.0  ...   65.953      NaN      NaN      N
aN
5         NaN      NaN  107005.4  ...  174.987      NaN      NaN      N
aN
6         NaN      NaN  180237.0  ...  186.172      NaN      NaN      N
aN
7         NaN      NaN  213725.0  ...  209.815      NaN      NaN      N
aN
8         NaN      NaN  248735.4  ...  259.248      NaN      NaN      N
aN
9         NaN      NaN  553234.8  ...  280.740      NaN      NaN      N
aN
10        NaN      NaN       NaN  ...  335.485      NaN      NaN      N
aN
11        NaN      NaN       NaN  ...      NaN      NaN      NaN      N
aN
12        NaN      NaN       NaN  ...      NaN      NaN      NaN      N
aN
13        NaN      NaN       NaN  ...      NaN      NaN      NaN      N
aN
14        NaN      NaN       NaN  ...      NaN      NaN      NaN      N
aN
15        NaN      NaN       NaN  ...      NaN      NaN      NaN      N
aN
16        NaN      NaN       NaN  ...      NaN      NaN      NaN      N
aN
17        NaN      NaN       NaN  ...      NaN      NaN      NaN      N
aN
18        NaN      NaN       NaN  ...      NaN      NaN      NaN      N
aN
19        NaN      NaN       NaN  ...      NaN      NaN      NaN      N
aN
20        NaN      NaN       NaN  ...      NaN      NaN      NaN      N
aN
21        NaN      NaN       NaN  ...      NaN      NaN      NaN      N
aN
22        NaN      NaN       NaN  ...      NaN      NaN      NaN      N
aN
23        NaN      NaN       NaN  ...      NaN      NaN      NaN      N
```

```
aN
24          NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
25          NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
26          NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
27          NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
28          NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
29          NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
..          ...     ...       ...  ...     ...       ...      ...
...
652         NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
653         NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
654         NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
655         NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
656         NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
657         NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
658         NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
659         NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
660         NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
661         NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
662         NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
663         NaN     NaN       NaN  ...     NaN       NaN      NaN      N
aN
```

```
664        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
665        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
666        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
667        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
668        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
669        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
670        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
671        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
672        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
673        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
674        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
675        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
676        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
677        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
678        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
679        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
680        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN
681        NaN      NaN        NaN  ...      NaN        NaN       NaN       N
aN

    acashcr   acashwd abankcol  abankr   aothcr  good
0    1100.0    850.33      0.0  5353.5  117.095   Yes
```

1	NaN	975.25	NaN	NaN	119.959	NaN
2	NaN	1073.09	NaN	NaN	NaN	NaN
3	NaN	1196.33	NaN	NaN	NaN	NaN
4	NaN	1218.28	NaN	NaN	NaN	NaN
5	NaN	1347.11	NaN	NaN	NaN	NaN
6	NaN	1347.39	NaN	NaN	NaN	NaN
7	NaN	1354.94	NaN	NaN	NaN	NaN
8	NaN	1376.73	NaN	NaN	NaN	NaN
9	NaN	1401.89	NaN	NaN	NaN	NaN
10	NaN	1414.84	NaN	NaN	NaN	NaN
11	NaN	1459.22	NaN	NaN	NaN	NaN
12	NaN	1551.03	NaN	NaN	NaN	NaN
13	NaN	1556.60	NaN	NaN	NaN	NaN
14	NaN	1578.27	NaN	NaN	NaN	NaN
15	NaN	1610.48	NaN	NaN	NaN	NaN
16	NaN	1694.57	NaN	NaN	NaN	NaN
17	NaN	1703.31	NaN	NaN	NaN	NaN
18	NaN	1708.37	NaN	NaN	NaN	NaN
19	NaN	1711.08	NaN	NaN	NaN	NaN
20	NaN	1733.15	NaN	NaN	NaN	NaN
21	NaN	1759.76	NaN	NaN	NaN	NaN
22	NaN	1761.38	NaN	NaN	NaN	NaN
23	NaN	1804.35	NaN	NaN	NaN	NaN
24	NaN	1813.55	NaN	NaN	NaN	NaN
25	NaN	1859.34	NaN	NaN	NaN	NaN
26	NaN	1882.11	NaN	NaN	NaN	NaN
27	NaN	1883.35	NaN	NaN	NaN	NaN
28	NaN	1890.78	NaN	NaN	NaN	NaN
29	NaN	1908.53	NaN	NaN	NaN	NaN
..	...	...	...	...	...	...
652	NaN	13342.86	NaN	NaN	NaN	NaN
653	NaN	13356.61	NaN	NaN	NaN	NaN
654	NaN	13403.73	NaN	NaN	NaN	NaN
655	NaN	13534.25	NaN	NaN	NaN	NaN
656	NaN	13555.35	NaN	NaN	NaN	NaN
657	NaN	13581.84	NaN	NaN	NaN	NaN
658	NaN	13583.68	NaN	NaN	NaN	NaN
659	NaN	13651.90	NaN	NaN	NaN	NaN
660	NaN	13679.61	NaN	NaN	NaN	NaN

```
661      NaN  13709.23       NaN      NaN       NaN    NaN
662      NaN  13896.20       NaN      NaN       NaN    NaN
663      NaN  14070.63       NaN      NaN       NaN    NaN
664      NaN  14157.97       NaN      NaN       NaN    NaN
665      NaN  14236.30       NaN      NaN       NaN    NaN
666      NaN  14330.66       NaN      NaN       NaN    NaN
667      NaN  14488.41       NaN      NaN       NaN    NaN
668      NaN  14541.34       NaN      NaN       NaN    NaN
669      NaN  14567.82       NaN      NaN       NaN    NaN
670      NaN  14663.72       NaN      NaN       NaN    NaN
671      NaN  14856.92       NaN      NaN       NaN    NaN
672      NaN  15131.87       NaN      NaN       NaN    NaN
673      NaN  15539.40       NaN      NaN       NaN    NaN
674      NaN  15740.80       NaN      NaN       NaN    NaN
675      NaN  15774.42       NaN      NaN       NaN    NaN
676      NaN  15933.13       NaN      NaN       NaN    NaN
677      NaN  15968.89       NaN      NaN       NaN    NaN
678      NaN  16111.84       NaN      NaN       NaN    NaN
679      NaN  16301.23       NaN      NaN       NaN    NaN
680      NaN  16853.59       NaN      NaN       NaN    NaN
681      NaN  16958.03       NaN      NaN       NaN    NaN

[682 rows x 38 columns]
```

In [9]:

```
skewness = data.skew(axis = 0, skipna = True)
print(skewness)
```

```
cardwdln     4.163910
cardwdlt     4.058582
cashcrn      0.636199
cashcrt      1.066390
cashwdn      0.731939
cashwdt      1.132872
bankcolt     2.029932
bankcoln     1.557739
bankrn       1.435638
bankrt       1.338442
othcrn       1.628444
othcrt       1.416784
```

```
days          0.392476
age           0.005981
cardwdlnd     3.684165
cardwdltd     4.000457
cashcrnd     -0.254937
cashcrtd      0.522146
cashwdnd     -0.081042
cashwdtd      0.463648
bankcoltd     1.369774
bankcolnd     0.834618
bankrnd       0.734049
bankrtd       0.538220
othcrnd       2.357087
othcrtd       0.976754
acardwdl      1.890530
acashcr      -0.135474
acashwd       0.142676
abankcol      1.369148
abankr        0.702417
aothcr        0.064614
dtype: float64
```

In [10]:
```
data.kurt()
```

Out[10]:
```
cardwdln      20.123161
cardwdlt      18.624095
cashcrn       -0.342262
cashcrt        0.597204
cashwdn       -0.279915
cashwdt        0.905055
bankcolt       3.634594
bankcoln       1.189602
bankrn         2.074571
bankrt         1.666862
othcrn         3.375370
othcrt         2.443918
days          -1.133008
age           -1.142675
cardwdlnd     15.805437
```

```
cardwdltd    20.796351
cashcrnd     -1.100812
cashcrtd     -0.596214
cashwdnd     -0.026374
cashwdtd     -0.496219
bankcoltd     0.404313
bankcolnd    -1.306391
bankrnd      -0.296355
bankrtd      -0.508840
othcrnd       5.145844
othcrtd       2.372778
acardwdl      2.770576
acashcr      -1.563058
acashwd      -0.890314
abankcol      0.399974
abankr        0.072172
aothcr       -0.406529
dtype: float64
```

In [11]:

```python
variance= np.var(data)
print(variance)
```

```
cardwdln     1.677356e+01
cardwdlt     1.103139e+08
cashcrn      1.141370e+03
cashcrt      6.329418e+11
cashwdn      3.772677e+03
cashwdt      4.480284e+11
bankcolt     5.042403e+11
bankcoln     4.113916e+02
bankrn       2.953844e+03
bankrt       4.218773e+10
othcrn       7.049007e+02
othcrt       2.285248e+07
days         2.701922e+05
age          1.602324e+02
cardwdlnd    8.882872e-06
cardwdltd    6.044290e+01
cashcrnd     4.713879e-04
```

```
cashcrtd      3.081678e+05
cashwdnd      3.742386e-04
cashwdtd      1.780657e+05
bankcoltd     2.905476e+05
bankcolnd     2.274284e-04
bankrnd       1.238636e-03
bankrtd       1.704554e+04
othcrnd       1.549265e-04
othcrtd       5.462800e+00
acardwdl      1.355304e+06
acashcr       8.699910e+07
acashwd       1.287223e+07
abankcol      2.724186e+08
abankr        5.550360e+06
aothcr        3.060319e+03
dtype: float64
```

In [12]:
```
data.corr()
```

Out[12]:

	cardwdln	cardwdlt	cashcrn	cashcrt	cashwdn	cashwdt	bankcolt	bankcoln
cardwdln	1.000000	0.969358	0.051299	0.086964	0.137415	0.156321	0.085416	0.034144
cardwdlt	0.969358	1.000000	0.034949	0.075140	0.141988	0.178193	0.118402	0.060569
cashcrn	0.051299	0.034949	1.000000	0.889803	0.488988	0.381115	-0.579456	-0.600754
cashcrt	0.086964	0.075140	0.889803	1.000000	0.475116	0.585396	-0.521281	-0.560187
cashwdn	0.137415	0.141988	0.488988	0.475116	1.000000	0.765641	0.304794	0.278753
cashwdt	0.156321	0.178193	0.381115	0.585396	0.765641	1.000000	0.346663	0.212979
bankcolt	0.085416	0.118402	-0.579456	-0.521281	0.304794	0.346663	1.000000	0.915954
bankcoln	0.034144	0.060569	-0.600754	-0.560187	0.278753	0.212979	0.915954	1.000000
bankrn	0.095480	0.083838	0.249843	0.112396	0.322926	0.015702	0.089817	0.119874
bankrt	0.052113	0.046947	0.197951	0.146750	0.385476	0.175690	0.291194	0.297311
othcrn	0.111668	0.101166	0.429271	0.342540	0.720813	0.431607	0.126920	0.167526
othcrt	0.265802	0.272592	0.426964	0.475052	0.794735	0.726744	0.263696	0.196004

	cardwdln	cardwdlt	cashcrn	cashcrt	cashwdn	cashwdt	bankcolt	bankcoln
days	0.152418	0.150257	0.553839	0.443112	0.920871	0.582045	0.202935	0.226665
age	-0.062075	-0.061343	-0.026922	-0.035580	-0.047731	-0.060657	-0.017684	0.001724
cardwdlnd	0.938762	0.913378	-0.005114	0.040128	0.024174	0.084803	0.053928	-0.000565
cardwdltd	0.897577	0.938943	-0.022885	0.026522	0.027286	0.106786	0.089416	0.026795
cashcrnd	-0.019091	-0.038094	0.781610	0.726612	0.016557	0.085253	-0.747551	-0.780254
cashcrtd	0.033970	0.021408	0.676002	0.836595	0.109660	0.354265	-0.607322	-0.653441
cashwdnd	0.041342	0.054533	0.130442	0.297087	0.605549	0.689194	0.318418	0.219267
cashwdtd	0.080801	0.109966	0.076729	0.376091	0.278005	0.756205	0.272580	0.104132
bankcoltd	0.037684	0.073718	-0.666353	-0.587557	0.064305	0.194721	0.884568	0.792198
bankcolnd	-0.015402	0.012426	-0.709001	-0.643716	0.021488	0.055826	0.809842	0.879329
bankrnd	0.041023	0.032867	0.016822	-0.069216	-0.075591	-0.235618	-0.006482	0.012172
bankrtd	-0.017307	-0.019139	-0.049842	-0.042088	-0.051679	-0.090665	0.195896	0.181636
othcrnd	-0.015025	-0.025873	0.018920	0.007426	0.082231	0.013170	-0.011232	0.025565
othcrtd	0.221684	0.236971	0.067430	0.240202	0.240377	0.470767	0.187713	0.063053
acardwdl	0.593111	0.648803	-0.006711	0.074050	0.059589	0.194217	0.113608	0.036502
acashcr	0.072294	0.054720	0.617472	0.789791	0.100012	0.275126	-0.641350	-0.692020
acashwd	0.084853	0.118284	0.018733	0.338631	0.093904	0.636173	0.198429	0.039620
abankcol	0.037320	0.073108	-0.666472	-0.587557	0.063344	0.194154	0.883860	0.791403
abankr	-0.084915	-0.069973	-0.065852	0.035748	0.023869	0.176936	0.212530	0.138169
aothcr	0.274723	0.304480	0.016882	0.240308	0.184523	0.521005	0.243795	0.080862

32 rows × 32 columns

In [13]:
```
data.cov()
```

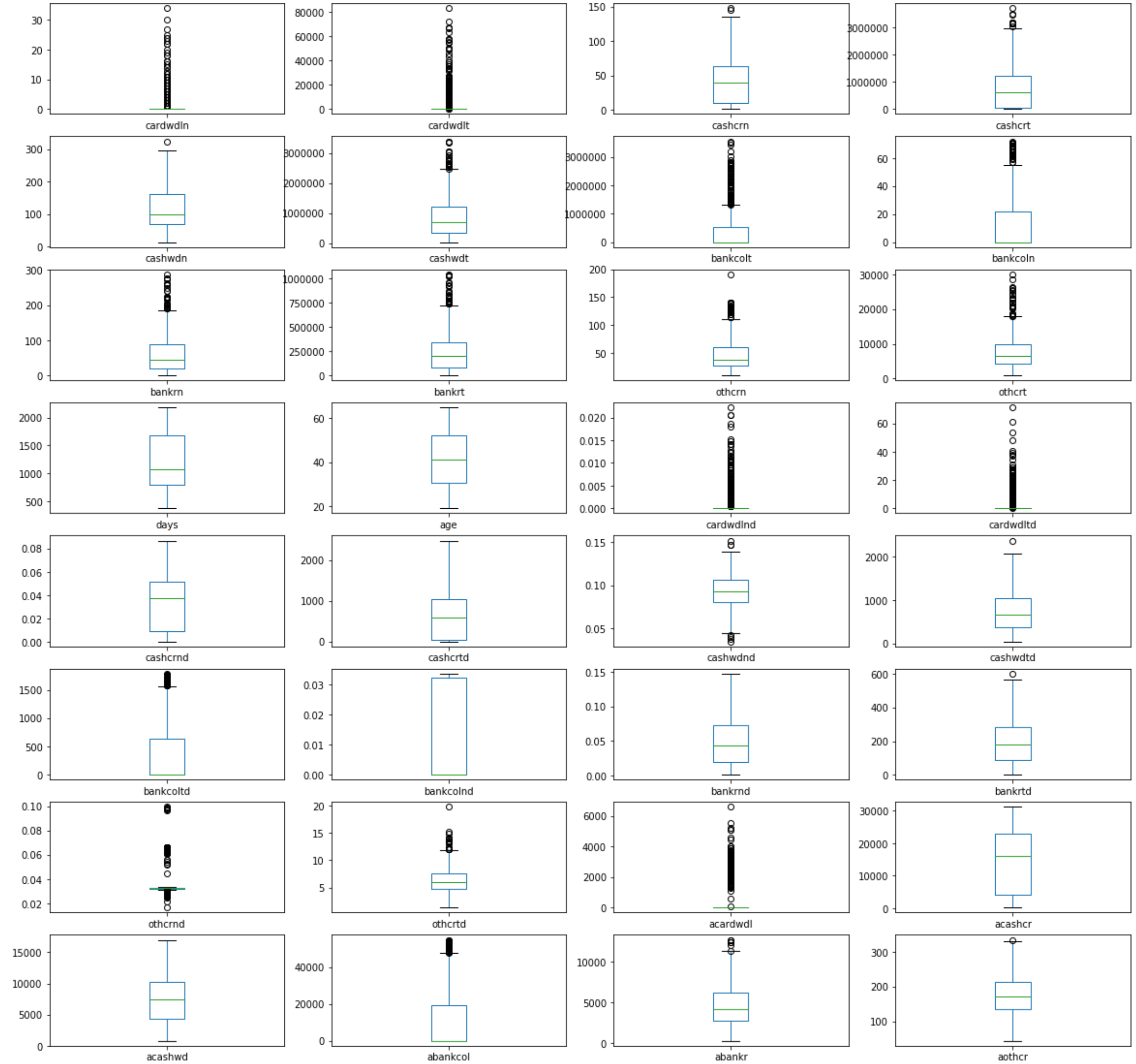
Out[13]:

	cardwdln	cardwdlt	cashcrn	cashcrt	cashwdn	cash
cardwdln	16.798190	4.175893e+04	7.108459e+00	2.837741e+05	3.461856e+01	4.291605e
cardwdlt	41758.928348	1.104759e+08	1.241932e+04	6.287861e+08	9.173355e+04	1.254574e
cashcrn	7.108459	1.241932e+04	1.143046e+03	2.395113e+07	1.016185e+03	8.630951e
cashcrt	283774.119996	6.287861e+08	2.395113e+07	6.338713e+11	2.325112e+07	3.121917e
cashwdn	34.618557	9.173355e+04	1.016185e+03	2.325112e+07	3.778216e+03	3.152388e
cashwdt	429160.475684	1.254574e+09	8.630951e+06	3.121917e+11	3.152388e+07	4.486863e
bankcolt	248775.049042	8.843629e+08	-1.392162e+07	-2.949235e+11	1.331333e+07	1.650125e
bankcoln	2.840480	1.292193e+04	-4.122631e+02	-9.052729e+06	3.477835e+02	2.895702e
bankrn	21.284208	4.792763e+04	4.594216e+02	4.867020e+06	1.079589e+03	5.720579e
bankrt	43902.802841	1.014267e+08	1.375630e+06	2.401536e+10	4.870257e+06	2.418969e
othcrn	12.160218	2.825223e+04	3.856076e+02	7.245943e+06	1.177196e+03	7.681437e
othcrt	5211.641148	1.370668e+07	6.905714e+04	1.809367e+09	2.336958e+05	2.328833e
days	324.953566	8.215301e+05	9.740256e+03	1.835138e+08	2.944405e+04	2.028072e
age	-3.222854	-8.167516e+03	-1.153024e+01	-3.588424e+05	-3.716559e+01	-5.146878e
cardwdlnd	0.011476	2.863384e+01	-5.156414e-04	9.528954e+01	4.431790e-03	1.694243e
cardwdltd	28.621614	7.678288e+04	-6.019662e+00	1.642874e+05	1.304912e+01	5.565153e
cashcrnd	-0.001700	-8.699517e+00	5.741555e-01	1.256929e+04	2.211255e-02	1.240761e
cashcrtd	77.346663	1.250050e+05	1.269674e+04	3.700225e+08	3.744596e+03	1.318293e
cashwdnd	0.003280	1.109649e+01	8.537733e-02	4.579066e+03	7.205856e-01	8.937296e
cashwdtd	139.847624	4.880916e+05	1.095468e+03	1.264453e+08	7.216126e+03	2.139045e
bankcoltd	83.314595	4.179588e+05	-1.215243e+04	-2.523353e+08	2.132153e+03	7.035782e
bankcolnd	-0.000953	1.971100e+00	-3.617593e-01	-7.734561e+03	1.993293e-02	5.643481e
bankrnd	0.005922	1.216697e+01	2.003129e-02	-1.940884e+03	-1.636458e-01	-5.558668e
bankrtd	-9.267697	-2.628331e+04	-2.201681e+02	-4.378091e+06	-4.150346e+02	-7.934777e
othcrnd	-0.000767	-3.387330e+00	7.967644e-03	7.364407e+01	6.295915e-02	1.098880e

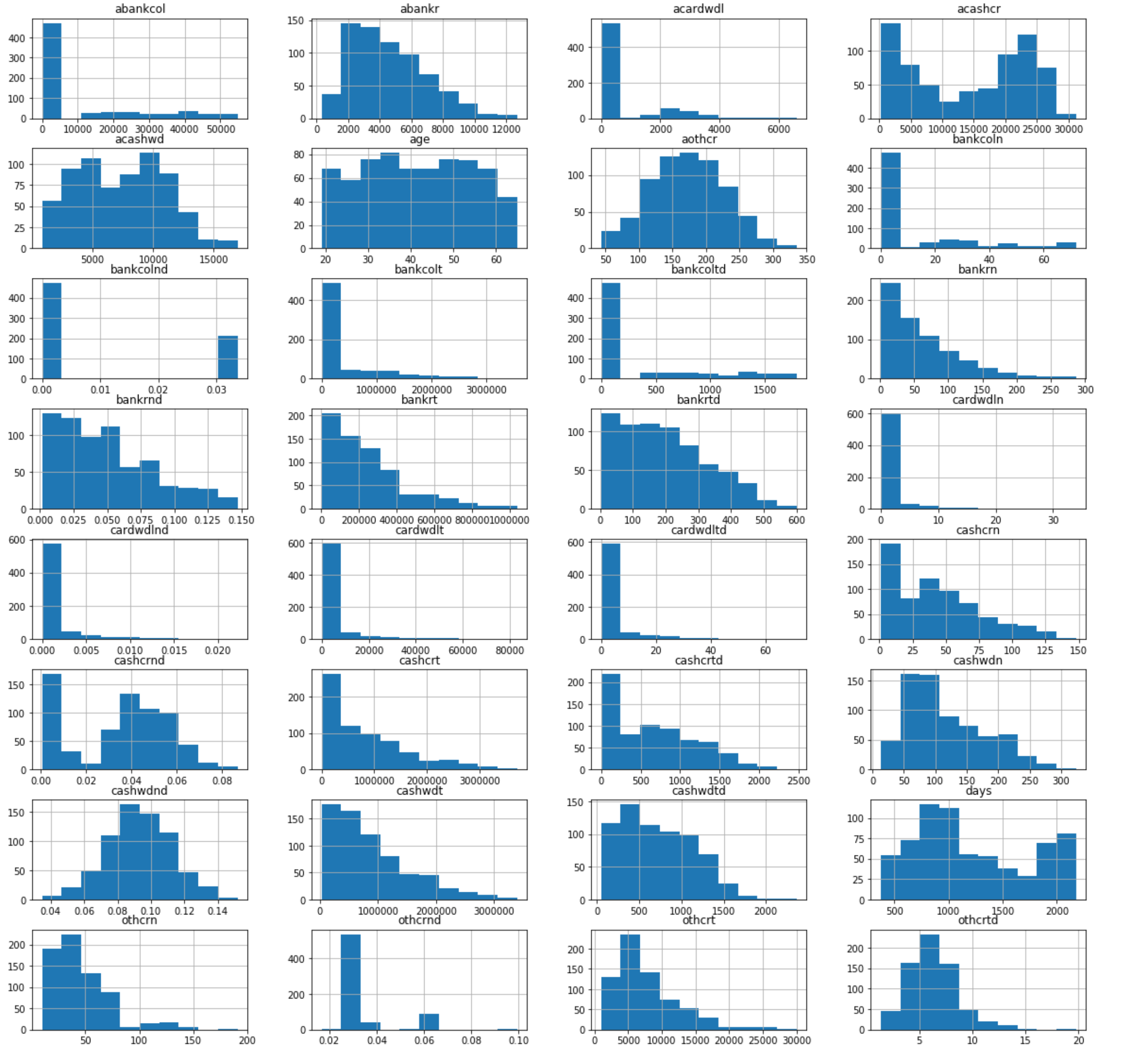
	cardwdln	cardwdlt	cashcrn	cashcrt	cashwdn	cash
othcrtd	2.125164	5.825791e+03	5.332245e+00	4.473038e+05	3.455917e+01	7.375710ε
acardwdl	2832.071948	7.944807e+06	-2.643237e+02	6.868503e+07	4.267246e+03	1.515641ε
acashcr	2765.732730	5.368517e+06	1.948612e+05	5.869329e+09	5.738149e+04	1.720201ε
acashwd	1248.661025	4.463812e+06	2.274001e+03	9.679952e+08	2.072403e+04	1.530003ε
abankcol	2526.469416	1.269221e+07	-3.721780e+05	-7.726585e+09	6.431118e+04	2.148103ε
abankr	-820.528614	-1.733974e+06	-5.249067e+03	6.710194e+07	3.459085e+03	2.794262ε
aothcr	62.334566	1.771715e+05	3.159856e+01	1.059181e+07	6.279063e+02	1.932036ε

32 rows × 32 columns

In [14]:
```
### VISULAIZATION OF THE DATA ###
data.plot(kind='box', subplots=True, layout=(16,4), sharex=False, share
y=False,figsize=(20, 40))
plt.show()
```

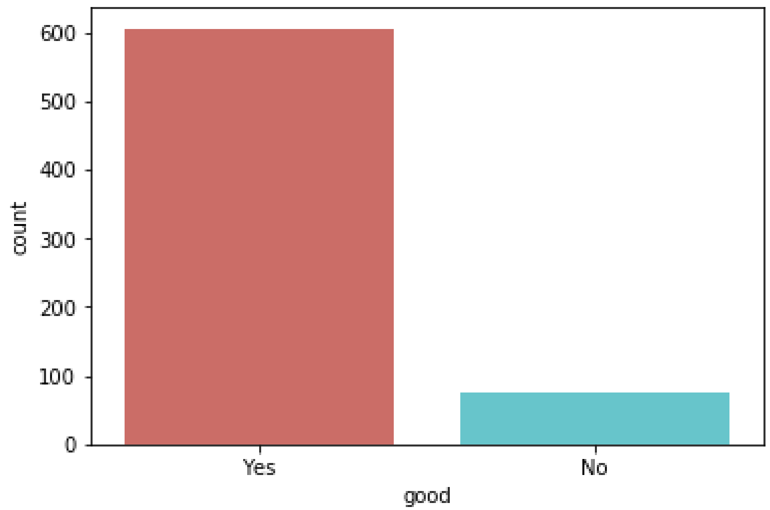


In [15]:
```
data.hist(layout=(16,4),figsize=(20, 40))
plt.show()
```



In [16]:

```
import seaborn as sns
sns.countplot(x='good',data=data,palette='hls')
plt.show()
```



OUTLIERS ARE PRESENT IN: 1)cardwdln 2)cardwdlt 3)bankcolt 4) bankrn 5)cardwdlnd 6)othcrnd 7)acardwdl 8)cashwdt 9) cardwdltd

In [17]:
```
print('Sex=',data['sex'].unique())
print('card=',data['card'].unique())
print('Second=',data['second'].unique())
print('Frequency=',data['frequency'].unique())
print('Region=',data['region'].unique())
print('good=',data['good'].unique())
```

```
Sex= ['M' 'F']
card= ['no' 'yes']
Second= ['Y' 'N']
Frequency= ['Monthly' 'Weekly' 'After_Tr']
Region= ['Prague' 'south Bohemia' 'north Moravia' 'east Bohemia' 'north
Bohemia'
 'south Moravia' 'west Bohemia' 'central Bohemia']
good= ['Yes' 'No']
```

In [18]:
```
data.dtypes
```

```
Out[18]: cardwdln       int64
         cardwdlt       int64
         cashcrn        int64
         cashcrt      float64
         cashwdn        int64
         cashwdt      float64
         bankcolt       int64
         bankcoln       int64
         bankrn         int64
         bankrt       float64
         othcrn         int64
         othcrt       float64
         days           int64
         sex           object
         card          object
         age          float64
         second        object
         frequency     object
         region        object
         cardwdlnd    float64
         cardwdltd    float64
         cashcrnd     float64
         cashcrtd     float64
         cashwdnd     float64
         cashwdtd     float64
         bankcoltd    float64
         bankcolnd    float64
         bankrnd      float64
         bankrtd      float64
         othcrnd      float64
         othcrtd      float64
         acardwdl     float64
         acashcr      float64
         acashwd      float64
         abankcol     float64
         abankr       float64
         aothcr       float64
         good          object
         dtype: object
```

In [19]:
```
X=data.iloc[:,0:37]
X.head()
```

Out[19]:

	cardwdln	cardwdlt	cashcrn	cashcrt	cashwdn	cashwdt	bankcolt	bankcoln	bankrn	
0	0	0	12	48400.0	237	1001191.0	1537936	70	89	55
1	0	0	17	45800.0	143	762135.7	741807	45	8	2
2	0	0	54	1488172.0	99	1215437.6	0	0	93	24
3	0	0	19	494494.0	51	375845.2	0	0	43	7
4	0	0	14	49700.0	37	156679.0	256060	17	45	10

5 rows × 37 columns

In [20]:
```
X.shape
```

Out[20]:
```
(682, 37)
```

In [21]:
```
X.columns
```

Out[21]:
```
Index(['cardwdln', 'cardwdlt', 'cashcrn', 'cashcrt', 'cashwdn', 'cashwd
t',
       'bankcolt', 'bankcoln', 'bankrn', 'bankrt', 'othcrn', 'othcrt',
'days',
       'sex', 'card', 'age', 'second', 'frequency', 'region', 'cardwdln
d',
       'cardwdltd', 'cashcrnd', 'cashcrtd', 'cashwdnd', 'cashwdtd',
       'bankcoltd', 'bankcolnd', 'bankrnd', 'bankrtd', 'othcrnd', 'othc
rtd',
       'acardwdl', 'acashcr', 'acashwd', 'abankcol', 'abankr', 'aothc
r'],
      dtype='object')
```

In [22]:
```
y=data.iloc[:,-1]
y.head()
```

Out[22]:
```
0    Yes
1     No
2    Yes
3     No
4    Yes
Name: good, dtype: object
```

SINCE WE HAVE CATEGORICAL VARIABLES IN THE DATA i.e., 1)sex 2)card 3)second 4) frequency 5) region WE ARE CREATING DUMMY VARIABLES

In [23]:
```
x = pd.get_dummies(X, columns=["sex", "card", "second", "frequency", "r
egion"], prefix=["sex", "card", "second", "frequency", "region",])
```

In [24]:
```
x.head()
```

Out[24]:

	cardwdln	cardwdlt	cashcrn	cashcrt	cashwdn	cashwdt	bankcolt	bankcoln	bankrn	I
0	0	0	12	48400.0	237	1001191.0	1537936	70	89	55:
1	0	0	17	45800.0	143	762135.7	741807	45	8	2(
2	0	0	54	1488172.0	99	1215437.6	0	0	93	24!
3	0	0	19	494494.0	51	375845.2	0	0	43	7!
4	0	0	14	49700.0	37	156679.0	256060	17	45	10

5 rows × 49 columns

In [25]:
```
x.columns
```

Out[25]:
```
Index(['cardwdln', 'cardwdlt', 'cashcrn', 'cashcrt', 'cashwdn', 'cashwd
t',
       'bankcolt', 'bankcoln', 'bankrn', 'bankrt', 'othcrn', 'othcrt',
'days',
```

```
        'age', 'cardwdlnd', 'cardwdltd', 'cashcrnd', 'cashcrtd', 'cashwd
nd',
        'cashwdtd', 'bankcoltd', 'bankcolnd', 'bankrnd', 'bankrtd', 'oth
crnd',
        'othcrtd', 'acardwdl', 'acashcr', 'acashwd', 'abankcol', 'abank
r',
        'aothcr', 'sex_F', 'sex_M', 'card_no', 'card_yes', 'second_N',
        'second_Y', 'frequency_After_Tr', 'frequency_Monthly',
        'frequency_Weekly', 'region_Prague', 'region_central Bohemia',
        'region_east Bohemia', 'region_north Bohemia', 'region_north Mor
avia',
        'region_south Bohemia', 'region_south Moravia', 'region_west Boh
emia'],
       dtype='object')
```

In [26]:
```
print(x)
```

```
     cardwdln  cardwdlt  cashcrn    cashcrt  cashwdn    cashwdt  bankco
lt  \
0           0         0       12    48400.0      237  1001191.0    15379
36
1           0         0       17    45800.0      143   762135.7     7418
07
2           0         0       54  1488172.0       99  1215437.6
0
3           0         0       19   494494.0       51   375845.2
0
4           0         0       14    49700.0       37   156679.0     2560
60
5           0         0        2     6800.0      164  1758719.0    22612
05
6           5     13200       35   635480.0       81   365494.2
0
7           0         0       38   867846.0       90   757291.9
0
8           0         0       14   252833.0       28   230921.2
0
9           5     14800       32   745897.0       79   515150.4
0
```

10	2	7500	47	1278771.0	105	1154757.0	0
11	0	0	95	1314120.0	176	663257.6	0
12	0	0	53	1475516.0	102	1259867.0	0
13	6	14500	33	484714.0	44	214448.2	0
14	1	3600	1	700.0	28	283509.8	388932
15	0	0	100	2092453.0	171	1611962.0	0
16	2	5900	33	504360.0	80	344350.4	0
17	0	0	22	426020.0	65	402199.0	0
18	0	0	4	8300.0	98	909638.0	1402089
19	0	0	74	1713632.0	138	1401114.0	0
20	0	0	26	616210.0	73	552032.0	0
21	2	6000	2	4000.0	98	1147291.0	1427611
22	0	0	67	939342.0	166	541517.6	0
23	0	0	89	1990266.0	207	1773345.8	0
24	0	0	32	177709.0	38	82944.4	0
25	0	0	73	1974944.0	114	1614008.8	0
26	0	0	89	1560877.0	129	853726.3	0
27	0	0	28	435438.0	67	252377.4	0
28	0	0	18	427443.0	24	269202.8	0
29	0	0	2	2400.0	86	1224322.0	14022

```
31
..          ...        ...       ...          ...       ...          ...
...
652           0          0         6      27100.0        49     166004.0     3658
82
653           0          0        66    1657264.0       205     978026.8
0
654           0          0       119    3150073.0       179    2390833.0
0
655          16      35000        31     410777.0        69     330850.4
0
656           0          0        70    1684174.0       146    1537978.6
0
657           0          0       118    3137980.0       239    2940206.0
0
658          11      33100         4       3300.0       225    2161662.0    32105
44
659           0          0       106    2548952.0       241    2311770.8
0
660           0          0        14      69400.0        71     283005.0     3787
90
661          11      31300         2       4000.0       135    1202778.2    18457
37
662           0          0        31     194900.0        72     286117.0     5658
96
663           0          0        71     896219.0       214     797634.4
0
664           0          0        52    1192751.0       139    1029923.0
0
665           4       8000         9      20100.0       145    1863768.8    20396
87
666           0          0        28     636472.0        58     374237.4
0
667           0          0       121    2816451.0       202    2305464.0
0
668           0          0        70    1066663.0       191     654712.2
0
669           8      17900        90    2702121.0       162    2550009.0
0
```

```
670          5     13200        45   889272.0        68   743488.0
0
671          2      3700         8    41700.0       159  1447077.4    20997
10
672          0         0         2     6800.0        76   788007.8    11160
73
673          0         0        44  1012868.0        72   694144.0
0
674          0         0        97   650086.0       170   367403.6
0
675          0         0        17    52300.0        80   165775.4     4769
35
676          0         0        36  1124913.0        70   934000.0
0
677          0         0        16   315374.0        33   203131.4
0
678          0         0        38   690261.0        78   608247.0
0
679          0         0         5    14800.0       136  1271219.0    19734
36
680         22     57400         2     4000.0       148  1105500.6    14992
77
681          0         0        43   680055.0       103   336301.8
0

     bankcoln  bankrn    bankrt  ...  frequency_Monthly  frequency_Week
ly  \
0          70      89  553234.8  ...                  1
0
1          45       8   20185.6  ...                  1
0
2           0      93  248735.4  ...                  1
0
3           0      43   75279.5  ...                  1
0
4          17      45  107005.4  ...                  0
1
5          50     116  480508.0  ...                  1
0
```

```
6            0     121  209642.0  ...                       1
0
7            0      11   81072.2  ...                       1
0
8            0       1    7348.0  ...                       1
0
9            0      64  147808.0  ...                       1
0
10           0      12   88442.4  ...                       0
0
11           0     240  595464.0  ...                       1
0
12           0      90  149581.6  ...                       0
1
13           0      24  215376.0  ...                       1
0
14          13       9   46166.5  ...                       1
0
15           0     160  455608.0  ...                       1
0
16           0      32  116999.0  ...                       1
0
17           0       1    1901.0  ...                       1
0
18          35      51  480121.0  ...                       0
0
19           0     126  242775.0  ...                       0
1
20           0       6   41448.0  ...                       1
0
21          28      58  188719.4  ...                       0
1
22           0     105  350907.0  ...                       1
0
23           0      22  171626.4  ...                       1
0
24           0      18   79512.0  ...                       1
0
25           0     127  329959.0  ...                       1
```

```
0
26         0    165  685481.2  ...              1
0
27         0     28  126479.0  ...              1
0
28         0     21  105903.0  ...              1
0
29        26     19  155803.8  ...              1
0
..       ...    ...       ...  ...            ...
...
652       22     76  207844.6  ...              0
1
653        0     94  633734.0  ...              1
0
654        0     88  722796.0  ...              0
1
655        0     21    6552.0  ...              1
0
656        0     24   77200.8  ...              1
0
657        0     48  154401.6  ...              0
1
658       70    191  950814.0  ...              0
1
659        0     22  154895.4  ...              1
0
660       29     38  119234.8  ...              0
1
661       47     79  579892.8  ...              1
0
662       35     59  426182.0  ...              0
1
663        0     49   70805.0  ...              1
0
664        0     22   54604.0  ...              1
0
665       38     12   98298.0  ...              1
0
```

```
666          0      39  200765.0  ...                    1
0
667          0     101  416002.0  ...                    1
0
668          0      93  368676.0  ...                    1
0
669          0      24   89872.8  ...                    0
1
670          0      12   44936.4  ...                    0
1
671         49      72  618062.4  ...                    1
0
672         23      49  269731.0  ...                    1
0
673          0      31  303148.5  ...                    1
0
674          0     146  261359.0  ...                    1
0
675         39     128  347096.9  ...                    1
0
676          0      31  104785.8  ...                    1
0
677          0      12   38133.0  ...                    1
0
678          0       5   23370.0  ...                    1
0
679         43      77  690478.0  ...                    0
1
680         51     104  273539.6  ...                    1
0
681          0     160  309438.0  ...                    1
0

     region_Prague  region_central Bohemia  region_east Bohemia  \
0                1                       0                    0
1                0                       0                    0
2                0                       0                    0
3                0                       0                    0
4                0                       0                    0
```

5	0	0	0
6	0	0	0
7	0	0	1
8	0	0	0
9	0	0	0
10	0	0	0
11	0	0	0
12	0	0	0
13	0	0	0
14	0	0	0
15	0	0	0
16	0	0	0
17	0	0	0
18	0	0	0
19	0	0	1
20	0	0	1
21	0	0	1
22	0	0	0
23	0	0	0
24	0	0	0
25	0	0	0
26	1	0	0
27	0	0	0
28	0	0	0
29	0	1	0
..	...	...	...
652	0	0	0
653	0	0	1
654	0	0	1
655	0	0	0
656	0	0	0
657	0	0	0
658	1	0	0
659	1	0	0
660	0	0	0
661	0	0	0
662	0	0	0
663	0	1	0
664	0	0	0

```
665         0           0          0
666         0           0          0
667         1           0          0
668         0           0          0
669         0           0          0
670         1           0          0
671         0           1          0
672         1           0          0
673         0           0          0
674         0           0          0
675         0           1          0
676         0           0          1
677         0           1          0
678         0           0          0
679         1           0          0
680         0           0          0
681         0           0          0

     region_north Bohemia  region_north Moravia  region_south Bohemia  \
0                       0                     0                     0

1                       0                     0                     1

2                       0                     1                     0

3                       0                     0                     1

4                       0                     0                     1

5                       0                     0                     1

6                       0                     1                     0

7                       0                     0                     0

8                       0                     0                     1

9                       1                     0                     0
```

10	1	0	0
11	0	0	0
12	0	1	0
13	0	1	0
14	1	0	0
15	0	0	0
16	0	1	0
17	0	0	1
18	0	0	1
19	0	0	0
20	0	0	0
21	0	0	0
22	0	0	0
23	0	0	0
24	0	1	0
25	0	1	0
26	0	0	0
27	0	0	0
28	0	0	0

29	0	0	0
..	...	...	...
652	0	0	0
653	0	0	0
654	0	0	0
655	0	0	0
656	0	0	1
657	0	0	1
658	0	0	0
659	0	0	0
660	1	0	0
661	0	1	0
662	0	0	0
663	0	0	0
664	1	0	0
665	0	0	1
666	0	0	0
667	0	0	0
668	0	1	0
669	0	1	0

670	0	0	0
671	0	0	0
672	0	0	0
673	1	0	0
674	0	0	1
675	0	0	0
676	0	0	0
677	0	0	0
678	0	0	0
679	0	0	0
680	0	0	0
681	0	1	0

```
      region_south Moravia  region_west Bohemia
0                        0                    0
1                        0                    0
2                        0                    0
3                        0                    0
4                        0                    0
5                        0                    0
6                        0                    0
7                        0                    0
8                        0                    0
9                        0                    0
10                       0                    0
11                       1                    0
```

```
12      0      0
13      0      0
14      0      0
15      1      0
16      0      0
17      0      0
18      0      0
19      0      0
20      0      0
21      0      0
22      1      0
23      1      0
24      0      0
25      0      0
26      0      0
27      0      1
28      1      0
29      0      0
..    ...    ...
652     0      1
653     0      0
654     0      0
655     0      1
656     0      0
657     0      0
658     0      0
659     0      0
660     0      0
661     0      0
662     1      0
663     0      0
664     0      0
665     0      0
666     0      1
667     0      0
668     0      0
669     0      0
670     0      0
671     0      0
```

```
672                        0                      0
673                        0                      0
674                        0                      0
675                        0                      0
676                        0                      0
677                        0                      0
678                        1                      0
679                        0                      0
680                        1                      0
681                        0                      0

[682 rows x 49 columns]
```

In [27]:
```
type(x)
```

Out[27]:
```
pandas.core.frame.DataFrame
```

In [28]:
```
x.shape
```

Out[28]:
```
(682, 49)
```

In [29]:
```
#Encode the y variable as well
from sklearn.preprocessing import LabelEncoder
labelencoder_y=LabelEncoder()
y_data_final=labelencoder_y.fit_transform(y)
y_data_final
```

Out[29]:
```
array([1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
       1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1,
1,
       1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1,
1,
       0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1,
1,
```

```
                    1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
          1,
                    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
          1,
                    1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1,
          1,
                    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0,
          1,
                    1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0,
          1,
                    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
          1,
                    0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1,
          1,
                    1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1,
          1,
                    1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1,
          1,
                    1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
          1,
                    0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1,
          0,
                    1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0,
          1,
                    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
          1,
                    1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
          1,
                    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1,
          1,
                    1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
          0,
                    1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
          1,
                    0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1,
          1,
                    1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
          1,
                    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
```

```
1,
        1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
1,
        1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
0,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1])
```

In [30]:
```
y_data_final.shape
```

Out[30]:
```
(682,)
```

DOING STEPWISE REGRESSION

In [31]:
```
import statsmodels.api as sm
X = pd.DataFrame(x)
y =y_data_final
def stepwise_selection(x, y,
                       initial_list=[],
                       threshold_in=0.01,
                       threshold_out = 0.05,
                       verbose=True):
    """ Perform a forward-backward feature selection
    based on p-value from statsmodels.api.OLS
    Arguments:
        X - pandas.DataFrame with candidate features
        y - list-like with the target
        initial_list - list of features to start with (column names of
 X)
        threshold_in - include a feature if its p-value < threshold_in
        threshold_out - exclude a feature if its p-value > threshold_ou
t
        verbose - whether to print the sequence of inclusions and exclu
```

```
sions
    Returns: list of selected features
    Always set threshold_in < threshold_out to avoid infinite looping.
    See https://en.wikipedia.org/wiki/Stepwise_regression for the detai
ls
    """
    included = list(initial_list)
    while True:
        changed=False
        # forward step
        excluded = list(set(x.columns)-set(included))
        new_pval = pd.Series(index=excluded)
        for new_column in excluded:
            model = sm.OLS(y, sm.add_constant(pd.DataFrame(X[included+[
new_column]]))).fit()
            new_pval[new_column] = model.pvalues[new_column]
        best_pval = new_pval.min()
        if best_pval < threshold_in:
            best_feature = new_pval.argmin()
            included.append(best_feature)
            changed=True
            if verbose:
                print('Add  {:30} with p-value {:.6}'.format(best_featu
re, best_pval))

        # backward step
        model = sm.OLS(y, sm.add_constant(pd.DataFrame(X[included]))).f
it()
        # use all coefs except intercept
        pvalues = model.pvalues.iloc[1:]
        worst_pval = pvalues.max() # null if pvalues is empty
        if worst_pval > threshold_out:
            changed=True
            worst_feature = pvalues.argmax()
            included.remove(worst_feature)
            if verbose:
                print('Drop {:30} with p-value {:.6}'.format(worst_feat
ure, worst_pval))
        if not changed:
```

```
            break
    return included

result = stepwise_selection(x, y)

print('resulting features:')
print(result)
```

```
C:\Users\srava\AppData\Local\Programs\Python\Python36\Lib\site-packages
\numpy\core\fromnumeric.py:2389: FutureWarning: Method .ptp is deprecat
ed and will be removed in a future version. Use numpy.ptp instead.
  return ptp(axis=axis, out=out, **kwargs)
C:\Users\srava\Anaconda3\lib\site-packages\ipykernel_launcher.py:33: Fu
tureWarning:
The current behaviour of 'Series.argmin' is deprecated, use 'idxmin'
instead.
The behavior of 'argmin' will be corrected to return the positional
minimum in the future. For now, use 'series.values.argmin' or
'np.argmin(np.array(values))' to get the position of the minimum
row.
```

```
Add  aothcr                         with p-value 1.26401e-14
Add  cashwdtd                       with p-value 5.04348e-21
Add  acashcr                        with p-value 1.66186e-06
Add  cashcrtd                       with p-value 6.74772e-10
Add  cashcrn                        with p-value 0.000715699
Add  second_N                       with p-value 0.000992668
Add  second_Y                       with p-value 9.1181e-40
Add  bankrn                         with p-value 0.00158976
resulting features:
['aothcr', 'cashwdtd', 'acashcr', 'cashcrtd', 'cashcrn', 'second_N', 's
econd_Y', 'bankrn']
```

In [32]:

```
### THESE ARE SIGNIFICANT VARIABLES ###
cols=(x[['aothcr','cashwdtd','acashcr','cashcrtd','cashcrn','second_N',
'second_Y','bankrn']])
x_final=cols
x_final.head()
```

Out[32]:

	aothcr	cashwdtd	acashcr	cashcrtd	cashcrn	second_N	second_Y	bankrn
0	153.107	469.16	4033.33	22.68	12	0	1	89
1	62.084	558.75	2694.12	33.58	17	1	0	8
2	221.432	1371.83	27558.74	1679.65	54	1	0	93
3	149.129	751.69	26026.00	988.99	19	1	0	43
4	147.371	307.21	3550.00	97.45	14	1	0	45

In [33]:

```
import statsmodels.api as sm
logit_model=sm.Logit(y,x_final)
result=logit_model.fit()
print(result.summary())
```

```
Warning: Maximum number of iterations has been exceeded.
         Current function value: 0.168058
         Iterations: 35
                           Logit Regression Results

=======================================================================
=======
Dep. Variable:                      y   No. Observations:
    682
Model:                          Logit   Df Residuals:
    674
Method:                           MLE   Df Model:
      7
Date:                Fri, 12 Jun 2020   Pseudo R-squ.:
0.5192
Time:                        14:48:20   Log-Likelihood:
-114.62
converged:                      False   LL-Null:
-238.37
Covariance Type:            nonrobust   LLR p-value:                 9.
438e-50
=======================================================================
=======
                 coef    std err          z      P>|z|      [0.025
```

```
0.975]
-----------------------------------------------------------------------
-------
aothcr          0.0606      0.007      8.622      0.000       0.047
  0.074
cashwdtd       -0.0060      0.001     -6.766      0.000      -0.008
-0.004
acashcr        -0.0003   5.54e-05     -4.954      0.000      -0.000
-0.000
cashcrtd        0.0043      0.001      3.660      0.000       0.002
  0.007
cashcrn        -0.0234      0.007     -3.222      0.001      -0.038
-0.009
second_N       -0.7734      0.684     -1.130      0.258      -2.114
  0.568
second_Y       26.3079   2.24e+05      0.000      1.000   -4.39e+05
4.39e+05
bankrn          0.0056      0.004      1.500      0.134      -0.002
  0.013
=======================================================================
=======

Possibly complete quasi-separation: A fraction 0.22 of observations can
be
perfectly predicted. This might indicate that there is complete
quasi-separation. In this case some parameters will not be identified.
```

```
C:\Users\srava\Anaconda3\lib\site-packages\statsmodels\base\model.py:51
2: ConvergenceWarning: Maximum Likelihood optimization failed to conver
ge. Check mle_retvals
  "Check mle_retvals", ConvergenceWarning)
```

Logistic Regression Model Fitting

In [34]:
```
xtrain, xtest, ytrain, ytest = train_test_split(
        x_final, y, test_size = 0.25, random_state = 0)
xtrain
```

Out[34]:

	aothcr	cashwdtd	acashcr	cashcrtd	cashcrn	second_N	second_Y	bankrn
556	149.053	472.15	16549.16	636.81	81	1	0	101
66	125.773	288.75	15455.61	549.73	61	1	0	140
571	149.172	895.22	22694.02	1078.53	96	1	0	89
299	107.867	170.87	7747.04	317.13	56	0	1	76
355	186.752	463.70	21477.87	745.08	23	1	0	23
626	72.609	148.53	10666.45	574.35	105	1	0	223
247	189.164	537.18	22853.92	878.31	49	1	0	164
624	143.352	123.68	6234.40	315.75	98	1	0	160
529	127.981	276.36	9765.80	561.53	46	0	1	45
609	277.244	2363.51	28540.93	2476.69	42	1	0	3
15	205.502	868.98	20924.53	1128.01	100	1	0	160
233	70.086	196.92	17939.38	295.30	16	1	0	54
215	184.912	722.57	1725.00	6.60	4	0	1	143
6	170.006	376.80	18156.57	655.13	35	0	1	121
268	173.933	887.00	1100.00	3.03	2	1	0	78
71	207.944	1556.60	2000.00	4.84	2	1	0	15
597	177.615	862.85	2625.00	12.98	4	1	0	106
294	115.059	97.93	6170.39	277.32	56	1	0	122
362	264.593	472.22	22829.85	823.37	66	0	1	79
650	188.012	129.55	8531.47	376.71	34	1	0	58
618	181.704	323.50	13448.42	544.37	88	0	1	104
188	195.589	630.46	23219.53	1080.45	53	1	0	82
570	235.717	1156.96	22463.48	1627.40	122	0	1	72
266	240.813	621.16	4225.00	22.56	4	0	1	12
354	62.612	221.78	5856.12	363.19	121	1	0	35

	aothcr	cashwdtd	acashcr	cashcrtd	cashcrn	second_N	second_Y	bankrn
90	157.935	267.77	13577.79	494.21	19	1	0	19
591	217.526	231.73	9957.28	468.37	54	0	1	63
681	110.939	286.46	15815.23	579.26	43	1	0	160
118	162.721	279.19	13073.29	462.11	31	1	0	71
521	130.108	514.59	22434.19	752.51	32	1	0	78
...	...	...	...	...	...	...	...	...
544	160.108	1066.93	23056.28	1200.19	57	0	1	20
639	168.729	538.82	4862.50	52.96	16	1	0	76
265	172.480	1049.54	25305.64	1144.07	42	1	0	12
288	226.704	1272.44	28314.08	1671.32	51	1	0	65
423	204.650	1376.00	27558.74	1679.65	54	1	0	93
147	188.927	549.19	4211.11	62.70	18	1	0	67
177	109.396	627.81	22292.52	702.98	52	1	0	16
99	179.813	432.56	16684.26	678.80	19	1	0	25
448	243.513	1286.83	23587.22	1404.15	114	1	0	48
431	138.989	91.48	5350.03	283.02	31	1	0	20
115	239.710	1122.14	400.00	0.65	1	0	1	32
72	200.380	478.73	20757.43	728.76	60	1	0	76
537	155.760	272.91	13369.96	607.04	40	0	1	106
672	229.204	1135.46	3400.00	9.80	2	1	0	49
174	199.109	509.81	19340.43	696.41	35	1	0	55
87	130.987	355.18	18603.24	685.28	68	1	0	201
551	214.495	836.99	22466.79	1302.57	125	1	0	143
486	139.006	215.99	10866.14	508.74	39	1	0	59
314	157.826	957.09	3200.00	11.48	3	1	0	9

	aothcr	cashwdtd	acashcr	cashcrtd	cashcrn	second_N	second_Y	bankrn
396	143.572	277.90	11003.97	445.18	39	1	0	93
600	204.551	1204.72	25542.77	1286.42	104	0	1	57
472	176.658	914.80	21906.58	1351.56	120	1	0	132
70	80.333	149.27	3920.00	23.44	5	0	1	58
599	216.148	1208.10	24093.28	1359.50	36	0	1	12
277	106.161	152.65	4217.60	216.29	48	1	0	22
9	198.010	574.30	23309.28	831.55	32	1	0	64
359	105.717	162.07	7542.30	307.58	23	1	0	28
192	263.953	1432.00	1950.00	7.65	2	1	0	52
629	234.408	770.06	25493.11	1007.46	46	1	0	25
559	155.088	526.67	16900.69	603.60	18	1	0	3

511 rows × 8 columns

In [35]:

```python
import statsmodels.api as sm
logit_model=sm.Logit(ytrain,xtrain)
result=logit_model.fit()
print(result.summary())
```

```
Warning: Maximum number of iterations has been exceeded.
         Current function value: 0.187870
         Iterations: 35
                           Logit Regression Results

======================================================================
=========
Dep. Variable:                      y   No. Observations:
      511
Model:                          Logit   Df Residuals:
      503
Method:                           MLE   Df Model:
        7
```

```
Date:                Fri, 12 Jun 2020   Pseudo R-squ.:
   0.4749
Time:                        14:48:20   Log-Likelihood:
  -96.002
converged:                      False   LL-Null:
  -182.83
Covariance Type:            nonrobust   LLR p-value:
4.269e-34
======================================================================
=========
                 coef    std err          z      P>|z|      [0.025
   0.975]
----------------------------------------------------------------------
---------
aothcr         0.0536      0.007      7.391      0.000       0.039
    0.068
cashwdtd      -0.0052      0.001     -5.739      0.000      -0.007
   -0.003
acashcr       -0.0003   5.79e-05     -4.481      0.000      -0.000
   -0.000
cashcrtd       0.0038      0.001      3.215      0.001       0.001
    0.006
cashcrn       -0.0178      0.008     -2.265      0.023      -0.033
   -0.002
second_N      -0.6044      0.759     -0.796      0.426      -2.092
    0.884
second_Y      34.8574   2.54e+07   1.37e-06      1.000   -4.97e+07
4.97e+07
bankrn         0.0048      0.004      1.197      0.231      -0.003
    0.013
======================================================================
=========

Possibly complete quasi-separation: A fraction 0.22 of observations c
an be
perfectly predicted. This might indicate that there is complete
quasi-separation. In this case some parameters will not be identifie
d.
```

```
C:\Users\srava\Anaconda3\lib\site-packages\statsmodels\base\model.py:51
```

```
2: ConvergenceWarning: Maximum Likelihood optimization failed to conver
ge. Check mle_retvals
  "Check mle_retvals", ConvergenceWarning)
```

In [43]:
```python
from sklearn.preprocessing import StandardScaler
sc_x = StandardScaler()
x_final_train = sc_x.fit_transform(xtrain)
x_final_test = sc_x.transform(xtest)
```

```
C:\Users\srava\AppData\Local\Programs\Python\Python36\Lib\site-packages
\sklearn\preprocessing\data.py:645: DataConversionWarning: Data with in
put dtype uint8, int64, float64 were all converted to float64 by Standa
rdScaler.
  return self.partial_fit(X, y)
C:\Users\srava\AppData\Local\Programs\Python\Python36\Lib\site-packages
\sklearn\base.py:464: DataConversionWarning: Data with input dtype uint
8, int64, float64 were all converted to float64 by StandardScaler.
  return self.fit(X, **fit_params).transform(X)
C:\Users\srava\Anaconda3\lib\site-packages\ipykernel_launcher.py:4: Dat
aConversionWarning: Data with input dtype uint8, int64, float64 were al
l converted to float64 by StandardScaler.
  after removing the cwd from sys.path.
```

In [44]:
```python
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(xtrain, ytrain)
```

```
C:\Users\srava\AppData\Local\Programs\Python\Python36\Lib\site-packages
\sklearn\linear_model\logistic.py:433: FutureWarning: Default solver wi
ll be changed to 'lbfgs' in 0.22. Specify a solver to silence this warn
ing.
  FutureWarning)
```

Out[44]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=
True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=0, solver='warn',
          tol=0.0001, verbose=0, warm_start=False)
```

In [45]:
```python
y_pred = classifier.predict(xtest)
```

In [46]:
```python
### CONFUSION MATRIX ###
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(ytest, y_pred)

print ("Confusion Matrix : \n", cm)
```

```
Confusion Matrix :
 [[ 11   6]
 [  2 152]]
```

WE HAVE 11+152= 163 CORRECT PRDICTIONS AND 2+6= 8 INCORRECT PREDICTIONS

In [47]:
```python
from sklearn.metrics import accuracy_score
print ("Accuracy : ", accuracy_score(ytest, y_pred))
```

```
Accuracy :  0.9532163742690059
```

In [48]:
```python
### PRECISION, RECALL AND F1-SCORE CALCULATION ###
from sklearn.metrics import classification_report
print(classification_report(ytest, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.85      0.65      0.73        17
           1       0.96      0.99      0.97       154

   micro avg       0.95      0.95      0.95       171
   macro avg       0.90      0.82      0.85       171
weighted avg       0.95      0.95      0.95       171
```

ROC CURVE

In [49]:
```
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
logit_roc_auc = roc_auc_score(ytest, classifier.predict(xtest))
fpr, tpr, thresholds = roc_curve(ytest, classifier.predict_proba(xtest)
[:,1])
plt.figure()
plt.plot(fpr, tpr, label='Logistic Regression (area = %0.2f)' % logit_r
oc_auc)
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.savefig('Log_ROC')
plt.show()
```