bank.R

91863

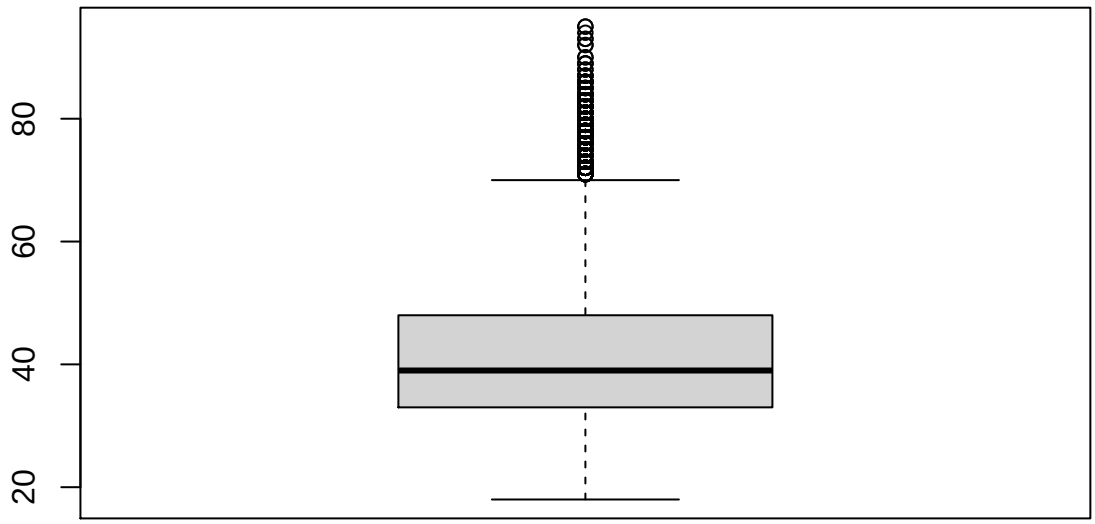
2021-01-31

```
############### data set information ######################
#The data is related with direct marketing campaigns of a Portuguese banking institution.
#The marketing campaigns were based on phone calls.
#Often, more than one contact to the same client was required, in order to access
#if the product (bank term deposit) would be ('yes') or not ('no') subscribed.

## dependent variable: y - has the client subscribed a term deposit? (binary: 'yes','no')

##we will do following steps
#1.import the data set
#2.remove the unnecessary column
#3.renaming the data set column
#4.check the data types,dim,summary
#5.density plot before preprocessing
#6.checking the NULL values(replace with continuous:mean,median or discrite:mode)
#7.check the outliers values(replace with continuous:mean,median or discrite:mode)
#8.overall distribution for all features(histogramplots)
#9.label encoding features(categorecal into numerical)
#10.check the correlation matrix(ggcorrplot)
#11.scale the data
#12.split the data train and test
#13.apply the machine learning models
#14.check the accuracy of each model
#15.deploy the high accuracy algorithm

#to remove the previous outputs and file in r studio
rm(list = ls())
#read the csv file
#bank=read.csv(file.choose())
bank=read.csv("customer_campaign.csv")
View(head(bank,10))

#renaming data set column
colnames(bank)[5]="default_credit"
colnames(bank)[7]="housing_loan"
colnames(bank)[8]="personal_loan"
colnames(bank)[9]="contact_type"
colnames(bank)[13]="current_compaign_contact_count"
colnames(bank)[14]="days_passed"
colnames(bank)[15]="previous_campaign_contact_count"
colnames(bank)[16]="previous_campaign_outcome"
```

```r
#checking the NULL values
#data.frame(colSums(is.na(bankData)))

if(length(which(is.na(bank)==T))){
  print("missing values are found")
}else{
  print("no missing values are found")
}
```

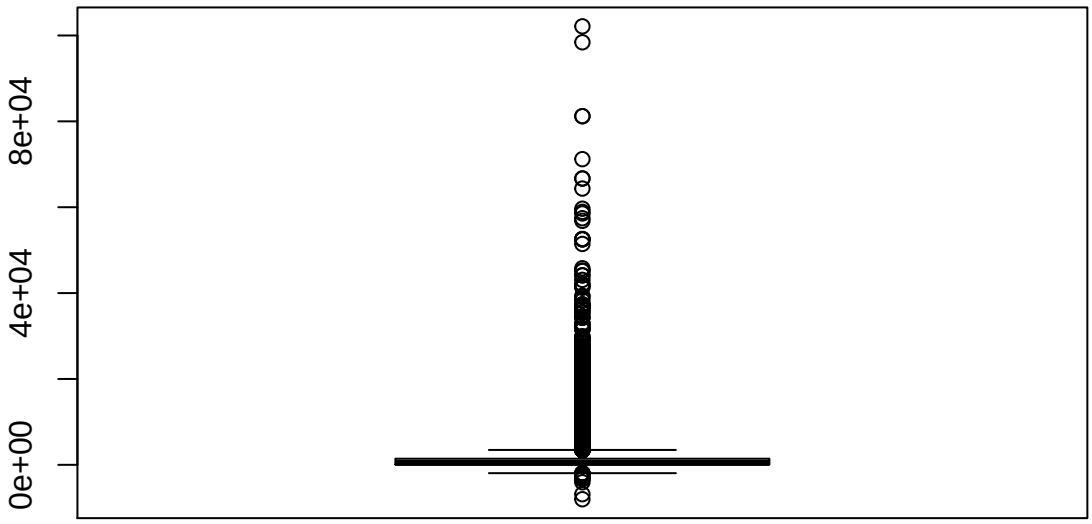
```
## [1] "no missing values are found"
```

```r
# check the outliers
unique(boxplot(bank$age)$out)
```




```
##  [1] 83 75 72 71 76 85 90 82 73 74 78 80 94 79 77 86 95 81 89 84 87 92 93 88
```

```r
unique(boxplot(bank$balance)$out)
```




```
##     [1] 10635  6530 12223  5935  4384  4080  5699 24598  8486  8730
##    [11]  4325 45248  5345  5435  3877  5431  5090  4070  4246  3754
##    [21]  6920  3659  7624  8823 58544  8266  3837  7444  5611 10350
##    [31]  5903  5956  4586  7317  6322  7601  5873  3932  4012  5034
##    [41]  4286  6259  5406  7459  4471 -3313  6809  3986  6411  4190
##    [51]  5694  9569  6248  4542  7816 18722  4785  3867  4509  6890
##    [61]  3675  5028  4128  6386 24299  3825  4646  7495  5521  3960
##    [71] 13308  6203  5356  4478  3839  8680  3571 26765  3478 12061
##    [81]  6958  8206  4436  3790 12482 29312  4265  9956  3635  3749
##    [91]  9004  4998  4871  3923  5154  5879 37378  7378  4666  6085
##   [101]  4135  4505  4151  8417 15801  7831  5700  7373  4446 23189
##   [111]  3641  3634  6874 11317 20718  6943  3652  4543  3658  3545
##   [121]  4248  3743  7735  7180  4906  8982 10749  7695 10600  5426
##   [131]  6050  6911  6439  4370  6532  4556  9131 15740  3630  5961
##   [141]  5678  6029  5291  4464  3625  5774  5131 56831  5006  4048
##   [151]  6127  4157  3649  9359  6659  5618  5041  3714  8150  5306
##   [161]  6510  4323  4582  3583 22370  9077  7864  3676  3820  7475
##   [171]  5788  6526  4928  9039  3990  3965  6060  9541  4580  4060
##   [181]  8016  4004 10399  4520 10576  4904  3501  4497  3840  5670
##   [191]  3657  7440  4015  3705  4541  8717  7727  6258  8837  5558
##   [201]  3689  4574  4168  3690 12269  3926 29887  3706  4391  4956
##   [211]  4166  6110  4537  4393  8318  3608 10561  8014  8167  7934
##   [221]  8180 12210  8806  4089  3736  4667  4301 12697 14752  5057
##   [231]  4378  6042  4357  6353  3722  6489  3855  4087  9714  3992
##   [241]  8195  3524  9051  4111  7449  4923  3776  6482  3544  4039
##   [251]  4438  6421  4482  4622  3750  7579  4635  6138  4648  4434
```

```
##  [261]  5499  3997  9374  4958 17983  8132  4170  5432  4011  8741
##  [271]  3763  4399  4897  5122  5260  8135 11697  5064  3559  5607
##  [281]  8291  6969  7171  3512  4798  5058 23867  4996  4963  7313
##  [291]  5436  4761  5143  7406  4822  5731  3514  8119  6102  3834
##  [301]  4025  8153  5754  4146  3954  8629  5024  3528  6982  4746
##  [311] 11512 35368  3773  7005  4315  5729  8784 13156 12961 15341
##  [321]  4465 11149  6217  5304 29050  5711  4499  4444  7198  5252
##  [331]  4665  5643  3468  4752  5853  6704  5366  3732  8874  7867
##  [341]  4924  4174  3744 24870  5527  4733  4441  4369  3696  3784
##  [351]  4191  5902  8541  8564  4243 11423  6714  3967  3517  6227
##  [361] 11315  4941  3798  4303  5547  5889 14692 17946 18347  9068
##  [371]  5745  7252  5553  4009  3767 17455  6374  4053  4519  3546
##  [381]  3643  3465  5205  5773  9194  5779 13849  5517  5505  4707
##  [391]  9874  3622  7598  4063  3888 19343  7544  9725  7216  5284
##  [401] 21111  4587  5888  5880  9894  4030  5151  4110  6575  4716
##  [411]  3532  6471  4131  3935  7137  6758  5455  4038  4388  3875
##  [421]  4793 16957 13186  6429 22867  7832  5891 10436  6164  9713
##  [431]  3723 10865  3620  3809  5172 10065  4513  8326  4681  5766
##  [441]  5222 16232  9779  6840  8594  4590  4050  3791  3672  6307
##  [451]  5980 16173  8298  4307  7063  5288 11417  3636  9636  3529
##  [461]  7304 22008  8089  5605  5214  4264  4366  5883  5336 22171
##  [471]  3955  3538  3753 11269  3557  6998 32948  6174 11953 15578
##  [481] 13761  6491 22755 10287 10045  5241  7098  7985  9687  8254
##  [491]  8839  6450  3670  5737  5801  5342  5310  8794 12956  4585
##  [501]  7712 16992  9767 -3372  6332  5244  9976  4692  4299  3499
##  [511]  6836  3516  3610  3886  6251 11804  5669  7606  5637  3629
##  [521] 11528  4425  3950  8131  4178 12855  8107  7707  3601  6695
##  [531]  5282  7290 14481  4660 11743  4414  3911  4539  5115  4120
##  [541]  7102  5724 25741  7108 -2049  6614  3674 12159  4791  7019
##  [551]  4136 12531 16063  8238  3693  5145  3587  5563  3782  5210
##  [561]  6312  3729 15511 18904  4143  3560  3487  3948  3472  6843
##  [571]  7505  4415  3574  5029  3868  4780  3561  4731  8558  6220
##  [581] 19797 12917  5164  8263 21096  6525  6321  4040  5195  7343
##  [591]  3496  5234  6285  4309 12282  6445  5149 11615  7634 13683
##  [601]  4466  9326 12926  3909  3862  3813  6141 16402  4408 12686
##  [611]  4788  6531  4736 17964  7858  3778 27733  3575  8938  7845
##  [621] 17891  6010  3844  4630 27359  3638  4084  3931  6290 11222
##  [631]  4737  8627  6281  7503 10787  7982  4409  5839  4013  4144
##  [641]  3611  6196  4281  3756 13052  5380 17036  4594  5278  4787
##  [651]  4096  7454  9121  3940  9824  8251  4209  6536  8590 14363
##  [661]  4046  3503 15442  9397 13360  4820  4207 11464  6397  7747
##  [671]  4318  4320  4458  7025 10855 21515  4579  3694  8562 12848
##  [681]  3924  3594  4674 21861  5506  3704  4150  9796  9059  6408
##  [691]  3823  5403  8127 22018  3595 45141  5462 11146 10360 19850
##  [701]  8001  8313  5267  4789  3598  5287 52499 10285  9192  3577
##  [711]  3653 21574  4830 12001  6020  5334 11671  5349  3783  7082
##  [721]  7067 16430  5098  8114 22928 10442  4639  5418  7179  9200
##  [731]  3952  7561  4037  7951  4874  4741  9618  6250  4362  5011
##  [741]  7027  4289  5768  5585 16431  3899 10483 14000  7372  7821
##  [751] 11016 13242  5201  4568  8592 22856 10179  6975  3558 10035
##  [761]  5473  4895  4222  3982 13698  5137  8412  7880  3644  5571
##  [771] 17413  4903  4997  3570  3819  3913  3467  4343  7783  5501
##  [781]  3812  7496  4844  8036 10333 16917  4056  4023 11516 13117
##  [791] 13893  4139  8368  8345 16843  7653  5523  7080  4840  8637
```

```
##  [801] 13089  8894 11084  4445  9019  7049  6204  8600  5092  7298
##  [811]  3671  4233  5560 10957 13551  8828  4800  6913  3664  4283
##  [821]  5305  5657  4397  3944  4138  3727  8379  5427  4848 18188
##  [831] 10638  5658  6317  4949  4176  4259  3628  7811  5543  5000
##  [841]  5908 11787  4723  3687  6553  7208  4232  5894  3708  5095
##  [851] 13292  6215  6072 21292 12180  4882  4359  5270  4641  3751
##  [861] 15352  4698 12581  4608  4075  3680  3764  7124  5050  5914
##  [871] 21664  6622  7938  5799  4722  7265  4696  5837 25824  6091
##  [881]  5559  5818  4322  4961  5251  7434  5830  6181  4266  5838
##  [891] 12877  4515  6613 22946  5279  3957  3942  7944  4073  7388
##  [901] 10465  5253  9231  8267  7798  3760 10721  3485 11265  5836
##  [911]  5048 11008  4930  5173  5348  6497 17361  5744  3768 17957
##  [921]  7419  4330  5233  3916  9405  4396 29125  5680 12130  4244
##  [931] 20541  4148  7451 10662  7509  6313  4112  3626  5874  4512
##  [941]  5087 11972  4045  4324  7177  5613  3737  6005  4204 10183
##  [951]  6449  4711  9711  3972  5735 13501  5109  3846  6495  4280
##  [961]  6025  7836  5736  6053  6835 -1968  4064  5614  4899  6955
##  [971]  3494 15477  5320  4365  4994  9689  3717 24055  5807 10984
##  [981]  9311  9103 27069  7123  3975  9669 10724  3728  3876  3832
##  [991]  5163 12186  6859 13494 12276  3556  3984  5052  6209  3854
## [1001]  6999  4655  5464  4777  4601  9246  9716  6512  7318  3895
## [1011]  8403  4134  5597  5078  3765 11285  4861 14902  4684  6298
## [1021] 10786 13054  4086  6360  3715  3492  5769 10236  3856 17339
## [1031]  4846  4145  4659  5804  3815  8652 21522  3759  8289  6207
## [1041]  4294  6100  5299  3988  7264  6618  4805  4385  4850  9301
## [1051]  3531 12245  6573  4044 13718  9444 -8019  3969 58932  5038
## [1061]  4688  4140  4314 12389  3604 51439  3540  8303 13229  4442
## [1071]  3493  3554  7711  5262  5068  4873  6350  4816  5758  4855
## [1081]  4488 19447  4693  6997  4380  3872  3800  6404  6253  3884
## [1091] -2093 -1965 10152  4453  3885  4054  9002  4597 15437  9994
## [1101]  5127  8079  3527  6586 13160  3814  3841  3466  8300  6739
## [1111] -2282  7522  7752  3508  3864 10438  4353  8224  4389  4654
## [1121]  4104  5009  7051  4459  4177  6014  7699 38279  3998 12939
## [1131]  5705  3500  4387 12731  3919 14058  4909  7032  5639 14611
## [1141]  6507  4943  3530  6108 10005  5231  5152 10189  3981  4149
## [1151]  5442  9228  4824 21963  6324  4842  5691 -6847  5865  3795
## [1161]  6567  6687  6570 45789  3902  5169  4344  4758  6393  4279
## [1171]  5116  5275  9965  4331  4306  4790  4101 14004  4386 14054
## [1181]  5248 14930  6012  4443  3703  6388  5920  8262  5829  3677
## [1191]  8918  3761 22557  9664  3510  4718  7612  5567  8873  4612
## [1201] 11532 13930  5074  5261  5698 10122  5423  4185  5706  4006
## [1211]  4069  6475 10685  9630  5156 -2082  7041 15561  5533  3933
## [1221]  3994  4321  4763 13315  3542  6114  5624  5483  7547  7190
## [1231]  5176  7225  6710  9173  3947  9881  4675  6269  6200  5934
## [1241]  7254 24312  3726  4381  8605  4413  6554  4295  4647  8106
## [1251]  3858  9143  6657 14530  4968 10628  5073  5969 13265  5441
## [1261]  4481  7159  3615 11254 10621  6641  6535  5632  8226  4372
## [1271]  6486  3831  4576 11310  6671 15681  3721  4664  5486  4000
## [1281]  4194  5414  3536 25290 17410  5681 10347  6763  3842  4567
## [1291]  4617 20584 13818  4358  5317  4853  3936  4900  6187  4336
## [1301]  8758  3648  4487  4105  4460  4392  5059  4908  3549  5350
## [1311]  3770  7365  4656  7957  7441  4401  4697 -2827  6030  4929
## [1321]  5640  7154  4879  5850  6145  5331 10655  4153  8218  4196
## [1331]  5181  4439  5132  4917  4374  9585  4440 14190  4492  8973
```

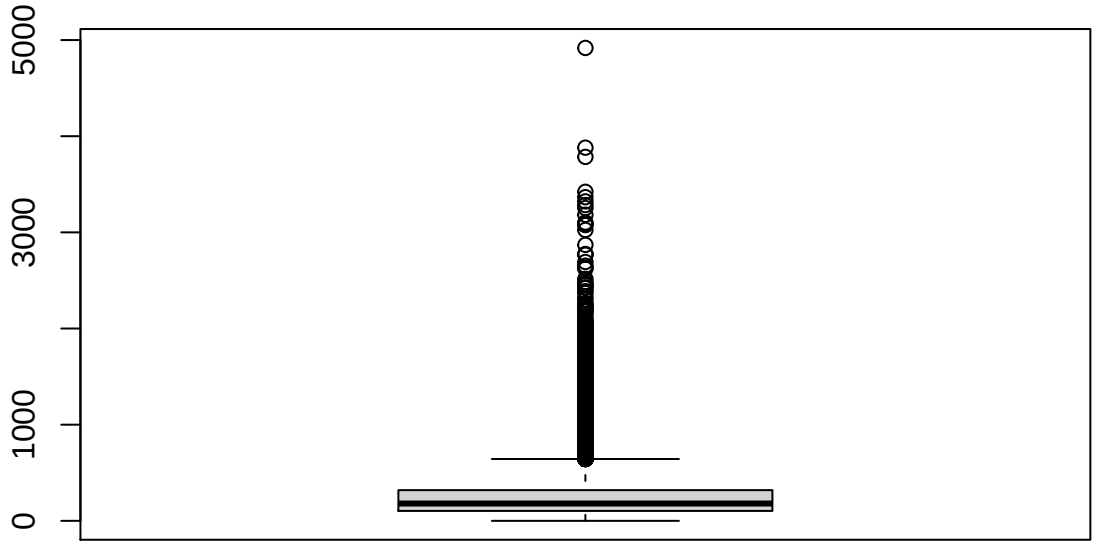
```
## [1341]  6878  3480  6699  5207 15515  9221  5223  4493 21244  5601
## [1351]  7337  4477 11391  7192  3794  3713  4837  6280  9683  5673
## [1361]  9720 11655 12438  4725  5361  7668  5094 16922 10888  5443
## [1371]  4424  7485 13118 14922  5953 13021  5704  7020  3766  4749
## [1381]  3718  4965  5988 24780 14282  6134 13189  3973  7788  9261
## [1391]  8279  6359 10757  4535 41923  5926 64343  4333  5003  5389
## [1401]  4043  6812 10077 11754  4765  5340  4953 10273  4108  4495
## [1411]  3469  5569  5999 14355  4235 15904  9680  4291  6822  6557
## [1421]  4719  3915  5239  5733  4547  4490  7303 15298 10950  5235
## [1431]  7487  6419  5102  5450  9608 66721  5204 42045  5177  5833
## [1441]  3590  8229  5550  6880 12928 10772  6961  8259 19348  3939
## [1451]  7138  9506  8422  4877  4843  9997  5581  5266  9304 21510
## [1461]  5016  3786  8844 10088 11971  3897  4596  6086  5108  4231
## [1471] 13338  8408  7513  3568  5665  3843 11797  7567  4889  3470
## [1481]  7876 23917  4123 11835 13834  4945  4186 -1980  8538  7742
## [1491]  4676  6493  3739  8351  3663  5219  9609 14093  3927  4782
## [1501] 15088  6753  3656  5904  5942  3533  7560  3616  8304 12409
## [1511]  8138 19213  8449  5907 11563  6004 23919  8897  4751  7586
## [1521]  3473  8553  3679  9634 11431  3665  3779  9622  4210  7353
## [1531]  6325  4720  3563 34646 14215 28433  6542  4158  6691  4367
## [1541]  7132  5887  8463 10052  4373  9146  9149  9827  9139  6170
## [1551] 10054 25947  4329  6413  3562  3481 13238 16377  6376  8332
## [1561]  8902  5574 13410 20479  6703  4610  3623  6945  3681  8029
## [1571]  4332  3579 10112  4022  4263  4561  3905  4455  6529  4016
## [1581]  4558  6284  6610  8163 31868 25856  4728 66653  8136  7197
## [1591]  6748  3586  4247  3818  5421  7296  4971 13763  4914  4564
## [1601]  8402  8121  5641  3797 13044  5124 41630  9202 23876 10639
## [1611] 14148  8139 11821  9804 12177  9207  3701  4772 17206 10500
## [1621]  6562  5399  7369  5495  4133  3655 12584  8413  5401  7135
## [1631]  3576 15302  5878  5561  4328  5060 22546  7649  7743  5004
## [1641]  3547  7631  8000 20794 10890  3486 52527  4948  3995  6619
## [1651]  7635  8097  7621 11305 15449 10215  4570  4636 23495  7907
## [1661] 13165  9346  5049  5809  4990 12766  4709  6264  4354  8971
## [1671] 26394  4028  3518  6271  3874  9135 11126  5188 43074  7134
## [1681]  6742 24450 29230  4213  3777  6144  4300  9864 17297  7073
## [1691]  7935  5535  7838 17555  4644  6343  9645  5167  4852  4687
## [1701] 10150 22815  7744  5910  7863  6563  3724  7084  6485  9324
## [1711]  6383  4979  5118  3918  5065  5976  7870  5943  3463  3762
## [1721] 20261  6743  6288  5603  7175  7881  6766  8047  4348  6171
## [1731]  6762  4450  3588  4522  4293 21614  3490  8480  3849  5936
## [1741]  8312  4577  7554  4695 13099 12519  9214  5303 24556  8749
## [1751]  7424  5346 26254  5514  5254 19268 10250  3537 12026  3912
## [1761]  7506  4683 12857  6490  9916  3585  4152  7028  8230  5847
## [1771]  4960  8648  6798  7162  4278  8969  5447  5381  5249 17432
## [1781]  4403  7387  5420  5827  5010  4962  7089 12226  5564 14107
## [1791]  3505 20187  6572  4230  4544  3735  5187  4826  4005 17092
## [1801] 16264  7007  9678  4583 22341  4504  6274  5326  4593 34247
## [1811] 11650  3552  3640  7595 12839  3684 10378  6153  7673 13342
## [1821]  7717 26575  4508  4819  9339  4982  4856 16517 13931  4463
## [1831]  8277  7530 14412  8491  6188  5862  4313  5811 10653  3520
## [1841]  7338  7848 18777 15520  3970 12845 16786  8142  6172 16397
## [1851] 11591  8563  3603 17655  5776  5343  6384 -2604 12569  6362
## [1861]  6000 35589 10031 11887 26172  4428  4589 -2122  5816  5037
## [1871]  5957 13853  4885  3572 14232  9072  5674 12539  9579  7623
```

```
## [1881]  5722 12437  6008  4432  3845  8781  3584  3564  5964  6043
## [1891] 12256  4717 10443 11766  4845  7118  3941  4649  4319  3889
## [1901] 16869 16358  4591  6286  6851  4411  4103  6427  4262 10218
## [1911]  6191  8647  4189 11246  4475  9102  5945  9347  8876 11524
## [1921]  6451  8038  4896  8093  8309  3711 11281  6929  6337  4480
## [1931] 18268  7558  8004  6077  3700  8863 11494  4395  3851 10995
## [1941]  8619  8583  5338  5498 26306  7249  5610  3953  3498 12675
## [1951] 12618  5848  6839  7800  3943 12737  8295  3495  5794  5238
## [1961]  4770  6933 10114  7396 12212 19358  6333  3780  6649  6921
## [1971] 12048  6964 17008  9895  3910 16486  4613 17769  9110 32464
## [1981]  8436  5296  8399  9269  8515  8979 10697  6767  7245 10041
## [1991]  6904  5393 22452 20600  5562 20527  4517  8649 15062 98417
## [2001]  4902  8953  5549  5972  6402  8489  9851  7104  6993  4420
## [2011] 15169  9252 20727 10395  5551  8545 11462  6994  6519  7541
## [2021] 13620  8077  3999  8859 12634  6909  9670 10021  7602  4341
## [2031]  7443 11675 29484  6922  5806  7962  8032  4121  4335  8892
## [2041] 20138  7100  8152  4223  5746  8957 10834  4269  9774  4888
## [2051]  6596 11386  3929  3651 20451 10386 13204 10735 14387  4091
## [2061] 16119 16489  5250  6754  3550  3827  5763 10142  8112  7773
## [2071] 12705  3605  7704  7928  3863  5007 10667  9911 17672 13836
## [2081]  6888  8460  6503 21854  7426  8785  3710  5039 17056  7429
## [2091]  7861 18558  3829 11350  3720  4760  6424  4894  5191  6410
## [2101]  6237  6981  7724  3817 14679  7696 16649  4581  9009 19796
## [2111]  8899 11839 10374  5461  7119 12270 13711  3803  5314  5075
## [2121]  3859  3774  6059  4712  7518  5206  7464  7242 18508  7741
## [2131]  4404  6150  7900 19391  4014  5133 14462  5045  4173 20580
## [2141]  4925  5276  9049  4254  8903  5220  8101  6889 11998  4094
## [2151]  4619 12132 14144 10406  4642  3938  8669  6316  7195  4117
## [2161]  6463  5368  3959  5626  5171 44128  4976  6107  6116 36935
## [2171]  5739  8043 41242  4536 27446 22196  5119 15474  9935  8689
## [2181]  5347  6873 17118 14440 16125 37176 19313  8381  4119  6080
## [2191]  4503 29207 13489  7384  6432 13015 20179  5301 13851 13354
## [2201] 20932  5784  5600  5293  7766  4402 24498  9047  9750  6900
## [2211]  7813 13578 57435 11287  4457  6157 16178  5129  5728  3816
## [2221]  6438 23663  7708 22520  4969  4713 13901 14522 29397 17924
## [2231] 16236  6385  7641  3534  9898  8017  6158  4847 36221  4418
## [2241]  6673 12704  4312  7482 22125 15120 15423 22569  3821  4809
## [2251]  5996  7825  6056 20798  6825  4764  5787  9319  6808  9531
## [2261]  8554  8265  8883  4932  4127  8025  3662  3945  5978  3869
## [2271]  4274  4557  5901  5715  8585  4130  4721  4638  4769  4959
## [2281]  3914 10536  4634  3578  5717  6683  5742 19833 17418  8141
## [2291]  5126  3962 10451  6857  7984  4317  7822  6745  5437  3519
## [2301]  3857  5701  6016  8990 15161  6236 13297  7289  5803 11093
## [2311]  5315  5372  6112  8626  5797  5061  8037  8118  5496 44134
## [2321] 23494  6178 19706  5082 11371 13565  3685  3850 14889  5885
## [2331]  4007  6162  4967  4831  3881 20772  7990  6132  3805  7408
## [2341]  6817 10613  4126  4082  7895  7351  4253 12495  3908  4383
## [2351]  4803  6915 19102  9160 17747  7114  4554 14344 13874  7357
## [2361]  7433  6684 26721  6106 14850  4588  3738  6101  5798  8104
## [2371]  5927  3977  6492 11298  5679 15261  4987 17609  5608 19985
## [2381] 26831  4198  5689 12114 11862  4991  3733  7702  4657  4700
## [2391]  5355  5359  4047 23878  7613 15459  3730  3917  5367 32685
## [2401]  4562 10180  8674  3698  6551  4808  4447  6422  4461  8929
## [2411]  5810  5157 10768  4886  8866  9329  5916  6651 10185 11385
```

```
## [2421]  6706  9601  9676  6468  7336 10469  5780 13408 -2712  4099
## [2431]  4066  4629  5417  5005  4910 10200  8666 11891  7263 11767
## [2441]  3740  5076  8860 17441  8623  5949 20806 12039 11555  4661
## [2451]  5043  4726  6533  5396  3752  5110  5467  5271 11066 13669
## [2461] 15485  5445  4394  6662  7685  4020 11387 11174  3612 21446
## [2471]  5542 13654  8311  6483  3771  9610  5012  6627  4137  5511
## [2481]  7048  5354  8444 13546  4795 -3058  5795  7107  4771  8148
## [2491]  3669  9305  7785  3688  4887  6637 16727 20011 17332  5452
## [2501]  4867 17335  4240  6700  3870  4095  6807 13562  8826 21024
## [2511]  7255  9698 11968 15841  6707  6677  6947 10086  3873 10925
## [2521]  9480  8556 10281 10072  3792  5193  7604  6770 20723  6791
## [2531]  6690  5583 11115  7345  7468  4003  5106  7791  3567  4859
## [2541] 31630  4287  3511  5341 11632  5802 27696  4041 36686 36252
## [2551] 10252  7780 20453  8947  4236 10907 10354  4984 11278  6368
## [2561]  4182  9367 10884  9407  9154  4229  6212  5872  4116  4430
## [2571] 10177  6392  3921  7620  9328  8339  6352  4835 25204  9262
## [2581] 10924  6882  5990  5289  6205  4860  6837  4239 11639  4031
## [2591] 34230  8509  5365  6242  8366 11757 11103  3806  9083  6815
## [2601]  9449  3507  7400  6346  9306  4500  5130  8044 18254  4714
## [2611]  5789 11686  7279 23552  4592  5709  4978  4786  5548  3551
## [2621] 15035  4986  3646 14657 10558  3589  4227 15030  3632  6737
## [2631]  4937  5918  5828 12392  8040  4406  6831  6182 20928  4778
## [2641] 28318  4527  8821  6089 11177  8548  5845  8536  6095  8654
## [2651]  7973  4954  7066 10357  4602  5781  3695  3573 10191  9277
## [2661]  4260 10910  6781  6979  4062  4974  5997  5854  4872  7531
## [2671] 16874 11854 13460 11752  4869  5792 -4057  4382  5089 10773
## [2681]  4575  5757 16563  5909  7103  4211 12198  6013 10133  9336
## [2691]  8950  5482 10269  4256 12607  6971  8094  3993  4841  7578
## [2701]  8023  5741 11219 13094  5086  5749  5091  8015  4792 11303
## [2711] 23076  3769  4311 21088 17023  3471  7818  4118 13107  4339
## [2721] 39098  3748  6968  3824 16873 10758  7968  4578  4920  5993
## [2731]  5215  4298 12322 27624  4545  7222 102127  5142 15445  5313
## [2741] 15787  8963  4132 22086  5265  9216  9756  6574  6246 15187
## [2751]  4216  5666 29184 10346  4694  4708 20585  8319 14533  6711
## [2761]  4200  8535  9902 10583  6590  5966 13164  9314  7781 19690
## [2771]  9629 23592  5946 11904  8603 18016  5084  3796  9883  8725
## [2781]  7929  4738  5973 26452  5312  7003  5474  5491  7879  4605
## [2791] 12767 19317  7386  7819  5539  4829  4297 10532  9064  6983
## [2801]  8103  6400  7050  5112 10861 52587 10394  4092  4290 20422
## [2811]  6991  7633 10373  6797  5805  8066 13014  6027 10776 18111
## [2821]  6844  4833 16432  4147  7218  4623 15265  7585 10171 12972
## [2831]  4083  6320 10905 10596  5871 10253  9962  8514  4068  6728
## [2841]  5584 14170  8494  5631 17875 11262  4985  7546 24277 59649
## [2851]  4129  6199  6850  4565 13450  9447  4744  7918  4451  6538
## [2861] 14220 13887 17739  7331  6046  4775 10943 39385 15834  6036
## [2871]  6784 26233 71188  7111  6879  8692  7469  3702  4017  4079
## [2881]  6481  3745 10889  7529 12067  4912  3810 24025  6746  4807
## [2891] 14646  5063  4572 16935 11240  7945  3951  3654  8278  7687
## [2901]  7458  7974  8729  8434  9317  4412 10332 12401 10541  9224
## [2911]  6279  4162  9366  4745  5695  4922  9421  3624 18881 13658
## [2921]  8334 12018  7105  6963 10788 81204  6513  4645  6447  6718
## [2931]  5021  7803  8919  3949  4727  3904 18967  7802 23421  4468
## [2941]  6571 37127  4448  4276  4606  8465 12980  6771  7010 29340
## [2951]  5196  7826  5958  5619  8750  7608  5236  7628  9001  3848
```

```
## [2961]  12264   5083   4599  26965   5861  15311   5047   7203   4305   7622
## [2971]   4531  12356   5008   4533  23047  29941   5397   4951  10971   5114
## [2981]   4071  29080   4024   5329   9299  31472   5475   4124  14968   5944
## [2991]  18931   3591   3504   6980  14352  13774   4680   8165   6403  17458
## [3001]   4416   7038   9710   8205  14204  16353
```

```
unique(boxplot(bank$duration)$out)
```




```
##   [1] 1666 1492  787 1778  812 1042 1467 1389  849  677 2033  673 1056  717  683
##  [16] 1077 1419  730  746  702  714  962  742  669  680  808  652 1201 1030  769
##  [31]  744  765 1623  678  699 1677  918 1297 1906  703  802  684  739 1597 1529
##  [46]  720  852  923  953  732 1521  800 1138  786  799  866 1581  650 1101  912
##  [61]  690 1062  688 2177  764 1273 1574  984 1689  697  944 1102  943  813 1040
##  [76] 1084  693 1119 1120  784  665  712 1007  667  982  756  807 2087  956  985
##  [91]  672 1187  826  847  659  772  929  710  705 2462  825  646  653 1028  654
## [106] 1087 1692 2016 1054 1170 1713  663 1080 1461  750 1178  752  878  834 1534
## [121]  836 1002  757 1147  820  788  832 1495  891 1083 1266  793 1727 1875  907
## [136]  723  704 1346 1386 3366 1000 2231 1167  806  766 1015  768 1001  845  853
## [151]  916  753  708  805  901  851 1052  647  771 1106  945  816 1721 1032  735
## [166]  942  824 1553 1328  686 1125  858  760  869  833  930  829  749  850  977
## [181]  927  762 1044  668  902  738 2241 1118 1423  747 1204 1013 1162  755  644
## [196] 1088 1036  695 1257 1165  651  920 1244  657  759  815  911  973  995 1224
## [211]  964 1156 1231 1051 1392 1867 1263  770  809  855  875  734  803  844  676
## [226]  656 1252 1143  731  754  679 1230  767  894  865 1340  897 1161  698 1128
## [241] 1135 1408  827 1193 1144 1023 1245 1064  882  792  798 1203 1022 1622  967
```

```
## [256]  886 1218 3078  661 1205 1882 1334  775 1777  774 1452 1376 1182 1045  999
## [271] 1063 1410 1287  843  919  777  725  719  692  905  783  872  958  648  951
## [286]  795  726  828  649 1091 1307  748  899  857  660 1681 1409  811  681 1697
## [301]  860 1094  965  987  671  935  713  700 1349 1171  736  785 1073  924  881
## [316]  691 1003  926  773  922  893 1438  937 1222 1034 1066 1099  682  974  745
## [331]  863  664 1560  880 1234  729  895  796  724  741 1272 1446  896  763  955
## [346]  674  740  776  751  709 3094  662 1168  938  861 1210  821 1183  675  694
## [361]  864 1730 1277 1196  733  791 1236 1207  936  932  689 1059  789  685 1611
## [376] 1185  814  859 1363 1109  645 2260  917  711  867 1133  854 1269  868 1097
## [391] 1500 1212 1343 1980  722  888 1075 1068  658  758  966 1576  707 1173  963
## [406]  941 1025  801 2456  959 1259  666 1516 1336 1242 1141 1519 1449 1181  908
## [421]  910 1149 1123 1009  701  761 1558 1053 1005 1865  817 1302 1018  884  718
## [436] 1121  939 1276 1994  862  968 1041 1288 2653 1085 1330 1469 1291 1055 1098
## [451]  940 1151 3881  952 1137  948 1159  706 1199  950  904 1290 1010  781  885
## [466]  993 1268 1243 1323 1093 1395 1238 1089 1021 1248  721 2769  986 1345 2621
## [481]  979 1208 1365  906  835 1029 1528 1487 1540 1812 1255 2093 1195 1060 2485
## [496] 1082  818  810 1318 2028 1017  779 1012 2635 1403  716  838 1573 1663 1617
## [511] 1478 1422  804  988  670  655  790 3183  856  780 1992  957  840 1434  782
## [526] 1103  889 1200 2201  830 1046 1767 1027  831 1720  819 1061  687 1111 1011
## [541] 1486  990  727 1014 1439 1426  998 1019  961 1341 2029 1499 1399  970 1973
## [556] 1649 1310 1397 1153 1130 1669 1271  778  794  981 1424 1142 1412  715 1806
## [571] 1150  873 1432 1039 1656 1339 1473 1275 1008 1584 1448 1390 1319 1175  728
## [586]  991  903 1303 1081  978 1503 1127  909 1360 1373  837  913 1139 1425 1105
## [601] 1877 1220 1342 1134  898 1352 1545 1833 1508 1237 1148 1037 1226 1608  870
## [616] 1152  874 1331 1327 1090 2078 1124 1309 1359  933 1417 1484 1441 1491  915
## [631] 1206 1869  823  947 1820 1602 1209 1946  743 2015 1031 1368  946 1369 1169
## [646] 1096  890 1076 1569 2516  797 1186 1058 1140 1078 1211 1567 2692  921 1107
## [661]  841 1739  846 1110 1488 1536 1092 1311 1113 1227  996 1357  931 2191 1249
## [676] 1250 1471 1456 1462 1834 1321 3422  876 1934 1306  994  848 1070 1687 1504
## [691] 1282  971 1650  696 1344 1613 1735 1476 1842  969  900  737  871 1314 1217
## [706] 3322 1184 1579 1871 1126 1364 1559 1293 1241 1740 1464 1532 1026 1258 1329
## [721] 1809 1374 1223 1033  976 1642 1176 2372 3253 2429 3284 1239 4918 1789 1606
## [736] 1531 1132  960 1978 1164 1855 1437  989 1377 1122 1554  925 1057 1548 1283
## [751] 1502 1065 1265 1792 1662 1468 1337 1435  997 1192 1816 1256 1490 1154 1035
## [766] 1166  972 1145 2420 1598 2453 1221 1158 1074  980 1571 1555 1067 1914 1048
## [781] 1387 1232 1972 1745 2150 1451 1600 1917  914 2770 1296 1431 3025 1776 1136
## [796]  928 1086 1790 1420 2256 1393 1635 1404 2053 1294 1416 1261 1381 1213 2775
## [811] 1916 1837 1823 1661 1960 1658 3102 1160 1047 1541 1971 1447 2330 3076 2870
## [826] 1859 1108 1353 2129 1190 1665 1691 1006 1594 1372 1038 1366 1174 1463 1095
## [841]  949 1332 1202 1112 1550 1817 1333 1348 1240 1100  954 1079  839 1131 1489
## [856] 1347 1957 1191  983 1254 1391  877  892 1262 1024 1049 1514 1114 1388 1925
## [871] 1710 1512 1966 1970 1975 1805 1279  992 1180 1020 1723 1129 1286 1380  934
## [886] 1326 1313 2301 1880 1460 2219 1361 1543 1603  883 1284  822  975 1225 1117
## [901] 2027 2055  879 1962 1702 1104 1551 1580 2187 1707 1233 2184 1628 1804 2062
## [916] 1472 1370 1616 1835 1563 2389 1407 1179 3785 1440 1405 1298 1246 1556
```

```
boxplot(bank$age,bank$balance,bank$duration)

#data has approx 45000 data point with 17 features.
dim(bank)
```

```
## [1] 45211    17
```

```r
#datatypes of column
data.frame(sapply(bank,class))
```

```
##                                 sapply.bank..class.
## age                                         integer
## job                                       character
## marital                                   character
## education                                 character
## default_credit                            character
## balance                                     integer
## housing_loan                              character
## personal_loan                             character
## contact_type                              character
## day                                         integer
## month                                     character
## duration                                    integer
## current_compaign_contact_count              integer
## days_passed                                 integer
## previous_campaign_contact_count             integer
## previous_campaign_outcome                 character
## y                                         character
```

```r
#summary of column
summary(bank)
```

```
##       age            job              marital           education
##  Min.   :18.00   Length:45211       Length:45211       Length:45211
##  1st Qu.:33.00   Class :character   Class :character   Class :character
##  Median :39.00   Mode  :character   Mode  :character   Mode  :character
##  Mean   :40.94
##  3rd Qu.:48.00
##  Max.   :95.00
##  default_credit        balance       housing_loan       personal_loan
##  Length:45211       Min.   : -8019   Length:45211       Length:45211
##  Class :character   1st Qu.:    72   Class :character   Class :character
##  Mode  :character   Median :   448   Mode  :character   Mode  :character
##                     Mean   :  1362
##                     3rd Qu.:  1428
##                     Max.   :102127
##  contact_type            day           month              duration
##  Length:45211       Min.   : 1.00   Length:45211       Min.   :   0.0
##  Class :character   1st Qu.: 8.00   Class :character   1st Qu.: 103.0
##  Mode  :character   Median :16.00   Mode  :character   Median : 180.0
##                     Mean   :15.81                      Mean   : 258.2
##                     3rd Qu.:21.00                      3rd Qu.: 319.0
##                     Max.   :31.00                      Max.   :4918.0
##  current_compaign_contact_count  days_passed    previous_campaign_contact_count
##  Min.   : 1.000                 Min.   : -1.0   Min.   :  0.0000
##  1st Qu.: 1.000                 1st Qu.: -1.0   1st Qu.:  0.0000
##  Median : 2.000                 Median : -1.0   Median :  0.0000
##  Mean   : 2.764                 Mean   : 40.2   Mean   :  0.5803
##  3rd Qu.: 3.000                 3rd Qu.: -1.0   3rd Qu.:  0.0000
##  Max.   :63.000                 Max.   :871.0   Max.   :275.0000
```

```
##  previous_campaign_outcome      y
##  Length:45211              Length:45211
##  Class :character          Class :character
##  Mode  :character          Mode  :character
##
##
##
```

```
#count analysis of categorical data
table(bank$job)
```

```
##
##        admin.   blue-collar  entrepreneur     housemaid    management
##          5171          9732          1487          1240          9458
##       retired self-employed      services       student    technician
##          2264          1579          4154           938          7597
##    unemployed       unknown
##          1303           288
```

```
table(bank$marital)
```

```
##
## divorced  married   single
##     5207    27214    12790
```

```
table(bank$education)
```

```
##
##   primary secondary  tertiary   unknown
##      6851     23202     13301      1857
```

```
table(bank$contact_type)
```

```
##
##  cellular telephone   unknown
##     29285      2906     13020
```

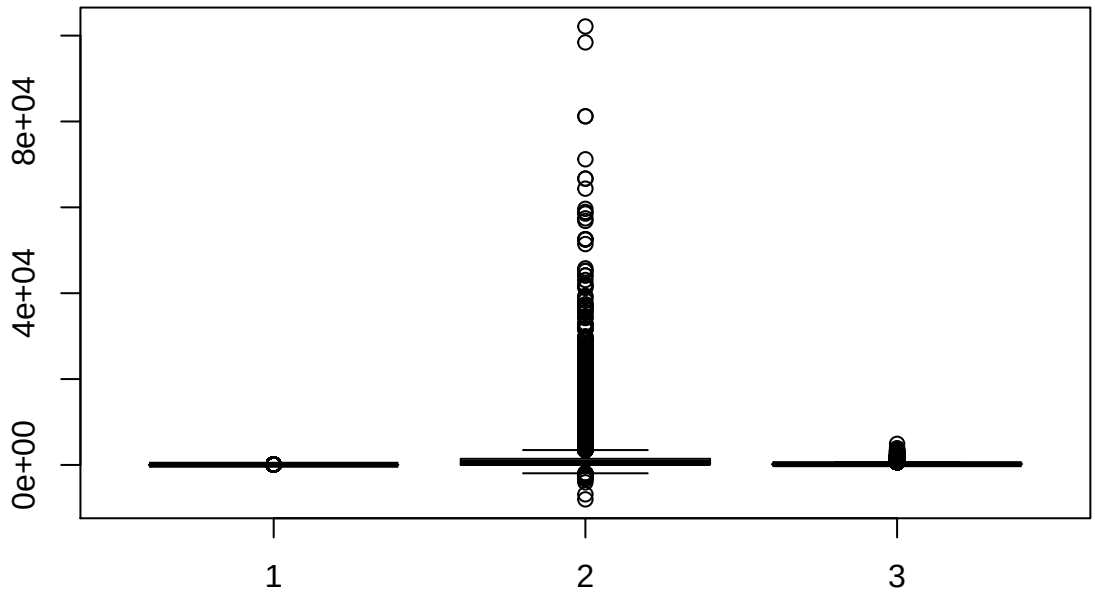
```
table(bank$month)
```

```
##
##   apr   aug   dec   feb   jan   jul   jun   mar   may   nov   oct   sep
##  2932  6247   214  2649  1403  6895  5341   477 13766  3970   738   579
```

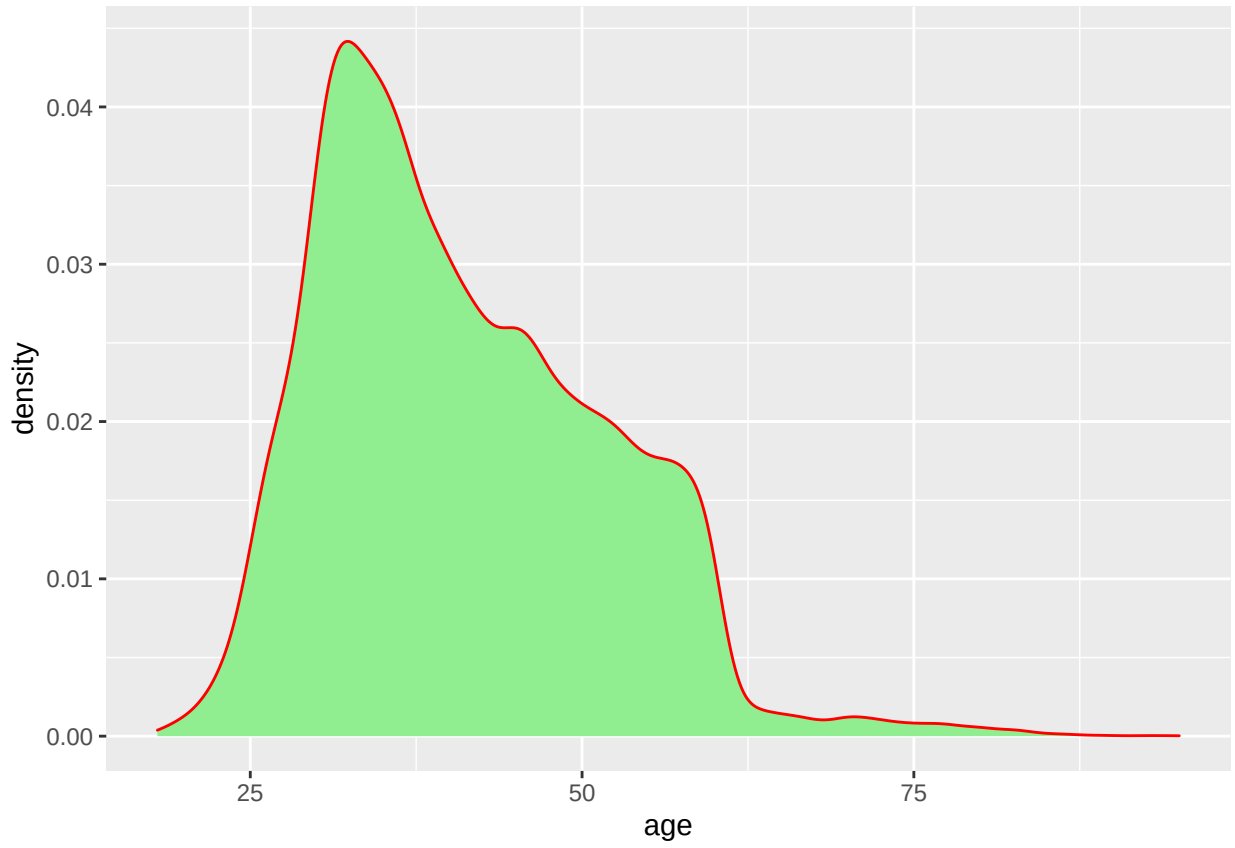
```
table(bank$previous_campaign_outcome)
```

```
##
## failure   other success unknown
##    4901    1840    1511   36959
```

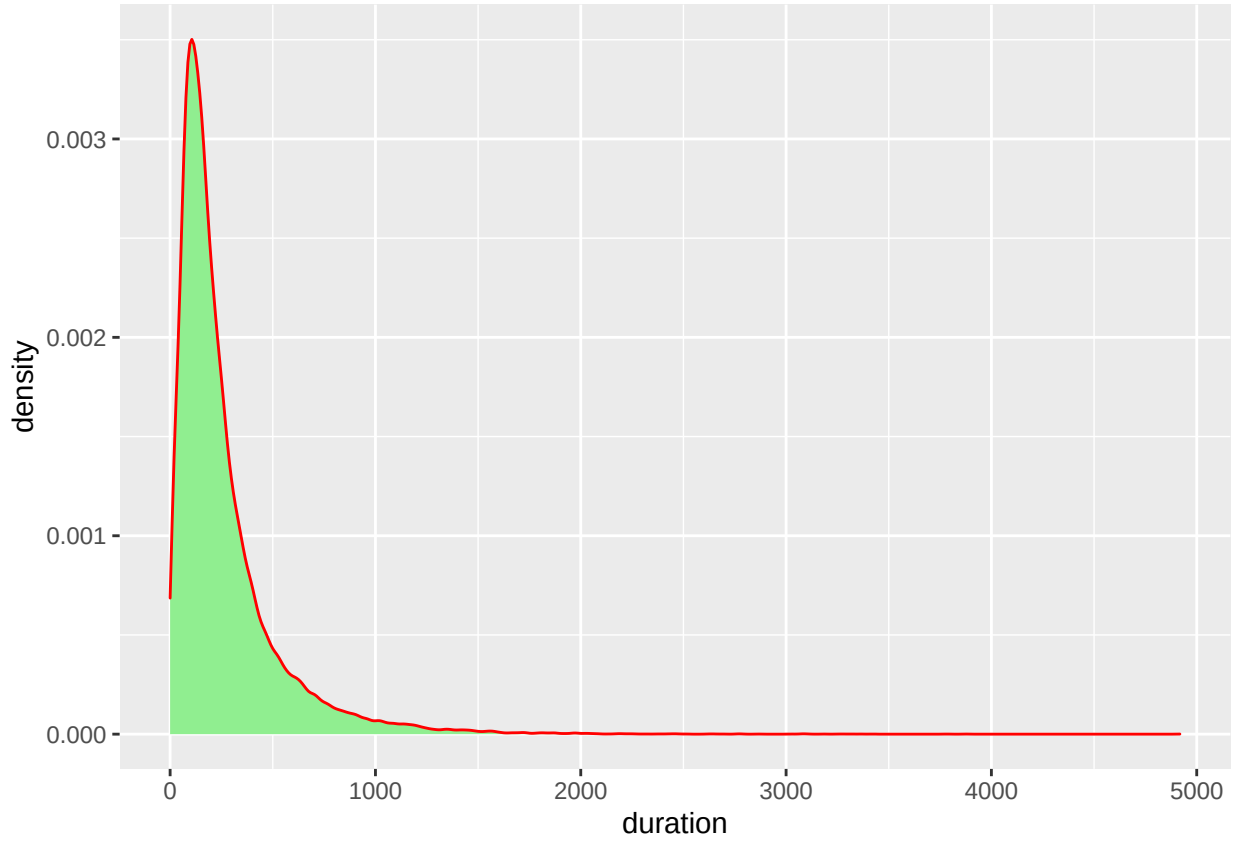
```r
#librarys
library(ggplot2)
```



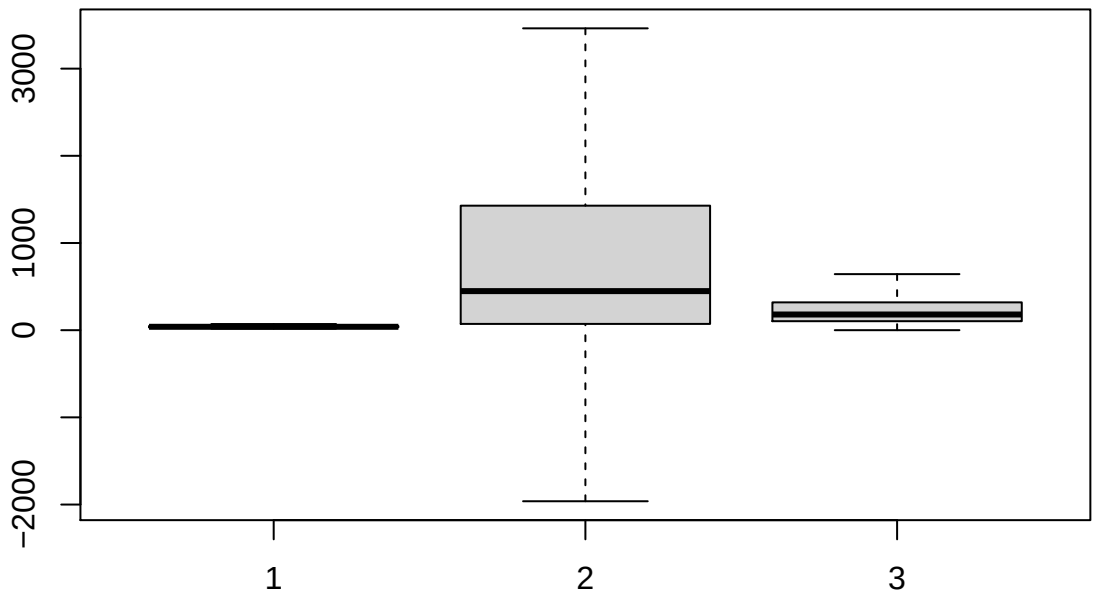

```r
library(rpart)
library(rpart.plot)
library(carData)
library(car)
library(class)
library(class)
library(lattice)
library(caTools)
library(caret)

#density plot before preprocessing
library(ggplot2)
ggplot(bank,aes(x=age))+geom_density(color="red",fill="lightgreen")
```




```
ggplot(bank,aes(x=duration))+geom_density(color="red",fill="lightgreen")
```




```
# outliers replace with mean

for (colName in c('age', 'duration','balance')) {
  high = quantile(bank[,colName])[4] + 1.5*IQR(bank[,colName])
  low = quantile(bank[,colName])[2] - 1.5*IQR(bank[,colName])
  for (index in c(1:nrow(bank))) {
    bank[,colName][index] = ifelse(bank[,colName][index] > high, high, bank[,colName][index])
    bank[,colName][index] = ifelse(bank[,colName][index] < low, low, bank[,colName][index])
  }
}


#after boxplot
boxplot(bank$age,bank$balance,bank$duration)
```



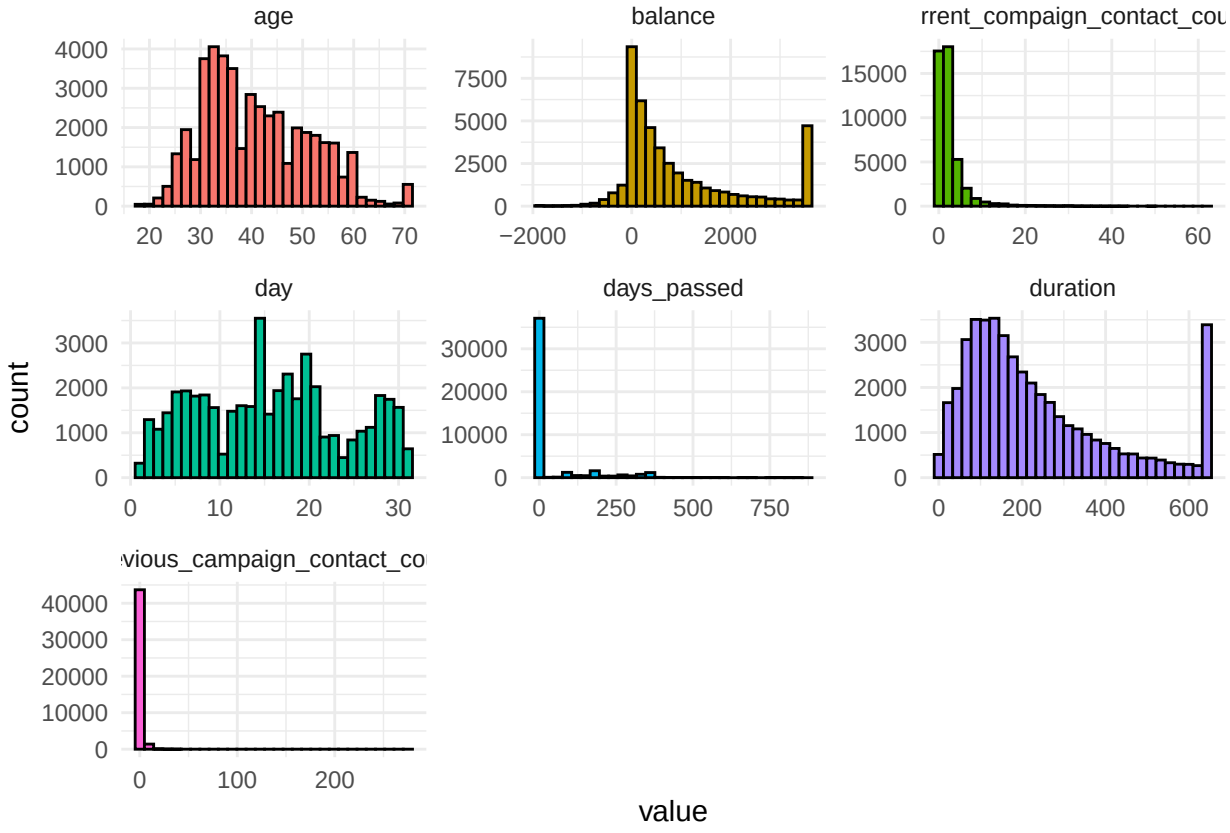

```
######Overall Distribution For All Features
library(patchwork)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0
## v purrr   0.3.4

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
## x dplyr::recode() masks car::recode()
## x purrr::some()   masks car::some()
```

```
bank %>%
  keep(is.numeric) %>%
  gather() %>%
  ggplot() +
  geom_histogram(mapping = aes(x=value,fill=key), color="black") +
  facet_wrap(~ key, scales = "free") +
```

```
  theme_minimal() +
  theme(legend.position = 'none')
```

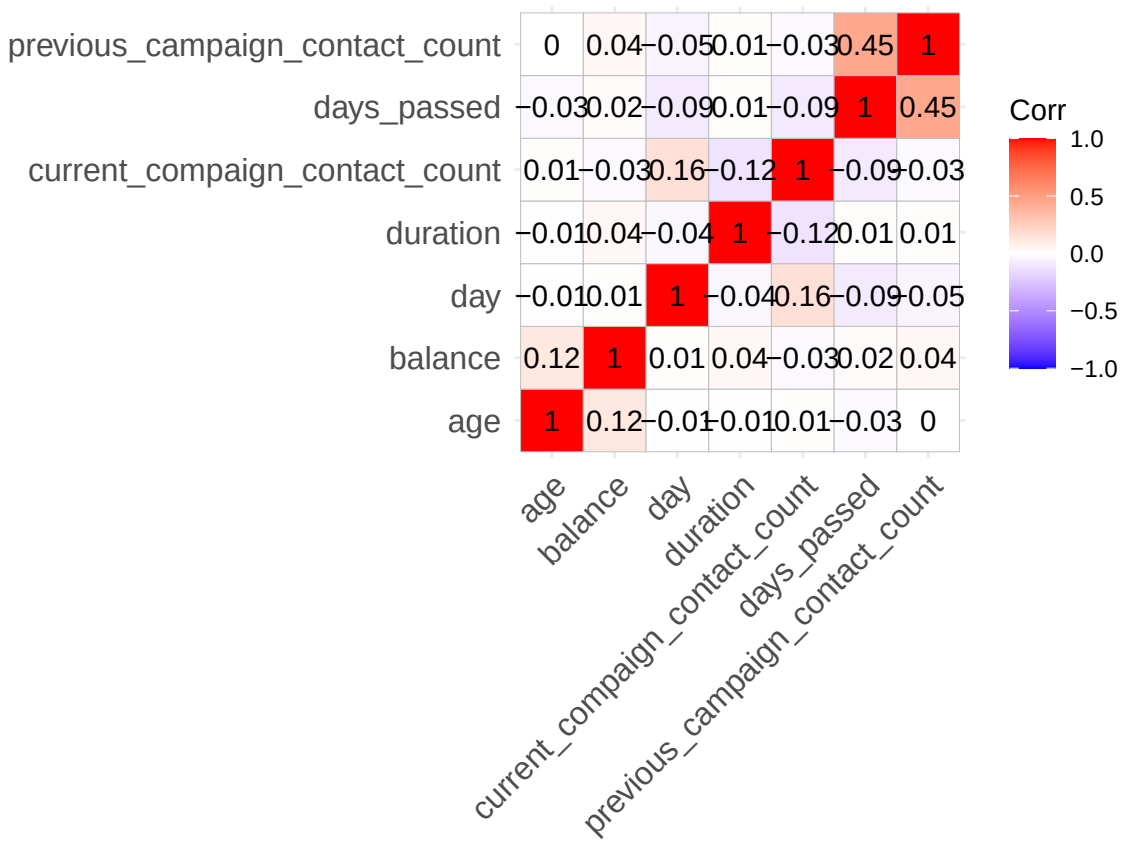
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



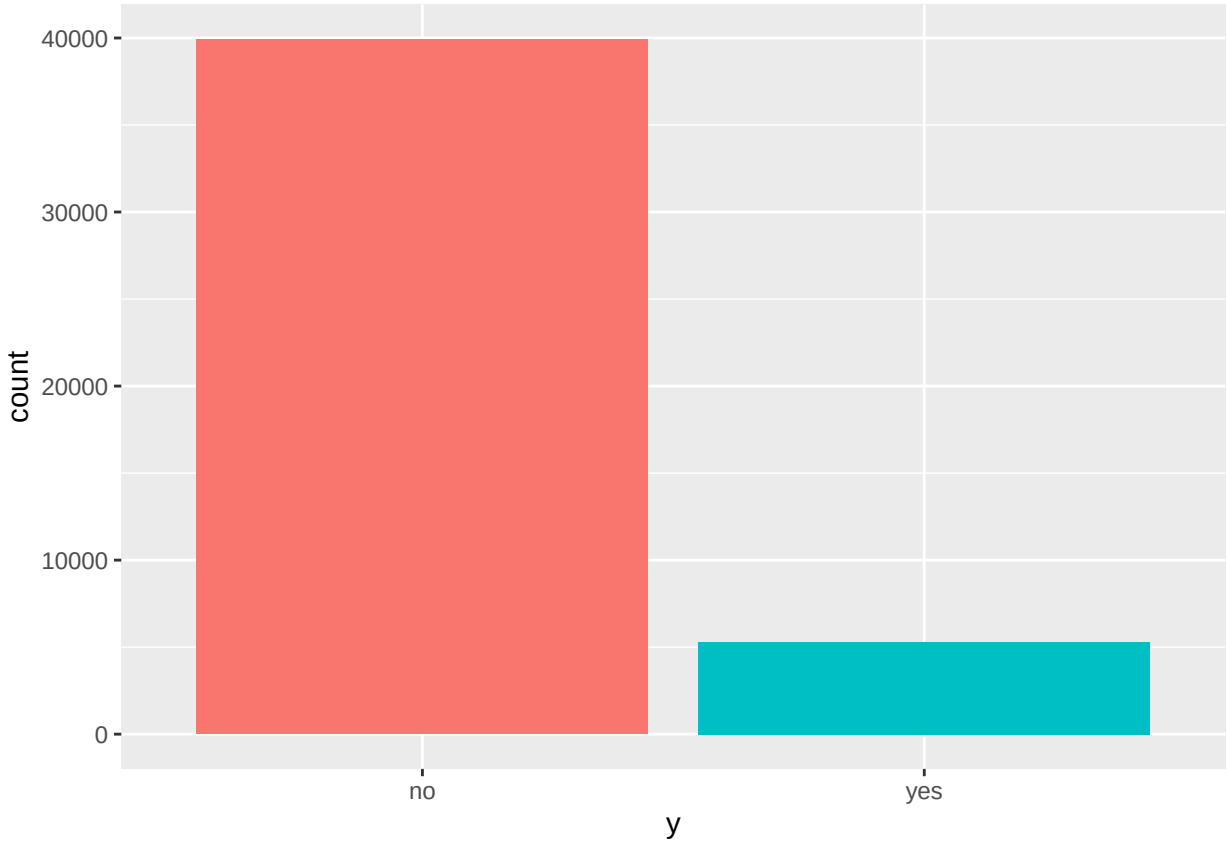

```
###Correlation Matrix

library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(ggcorrplot)
Name=names(which(sapply(bank, is.numeric)))
corr=cor(bank[,Name], use = 'pairwise.complete.obs')
ggcorrplot(corr, lab = TRUE)
```




```
###########Response Variable
ggplot(bank, aes(y, fill = y)) +
  geom_bar() +
  theme(legend.position = 'none')
```

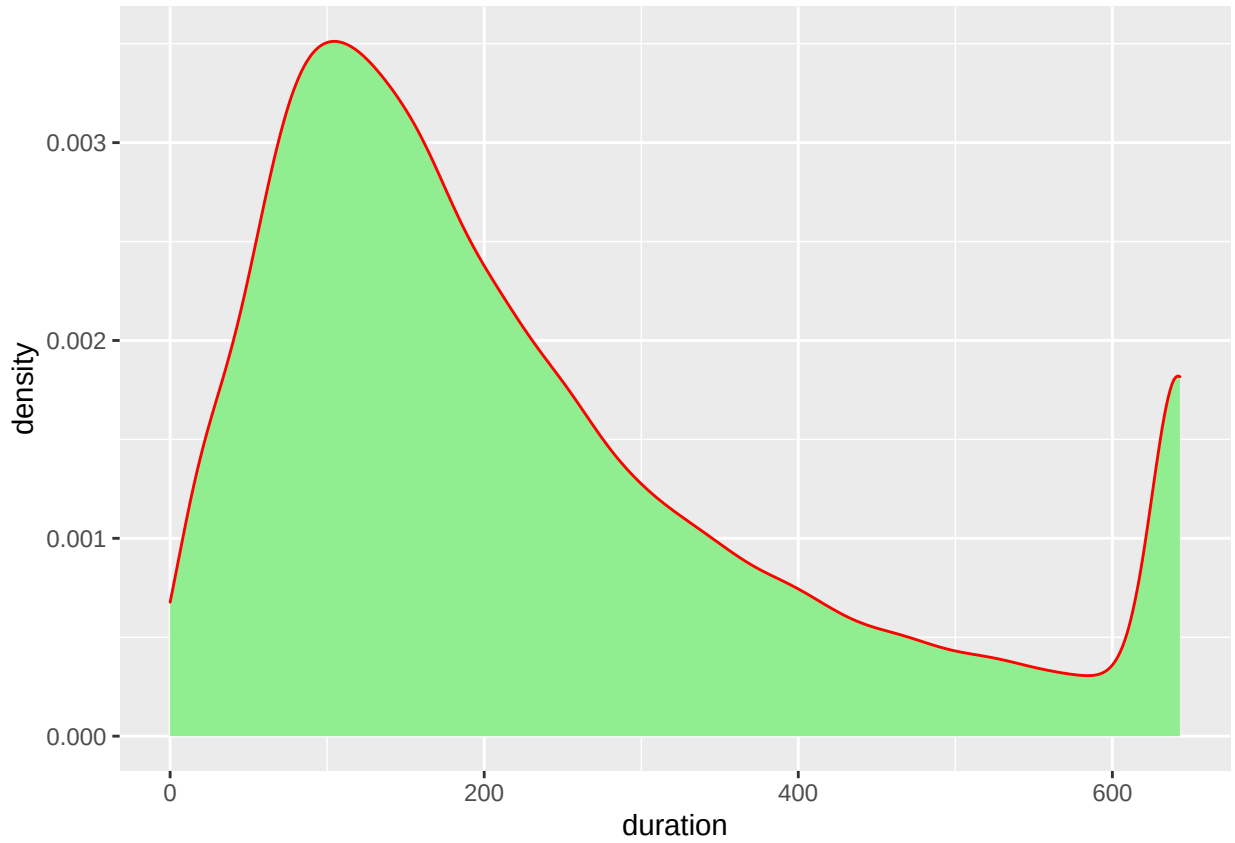



```
table(bank$y)
```

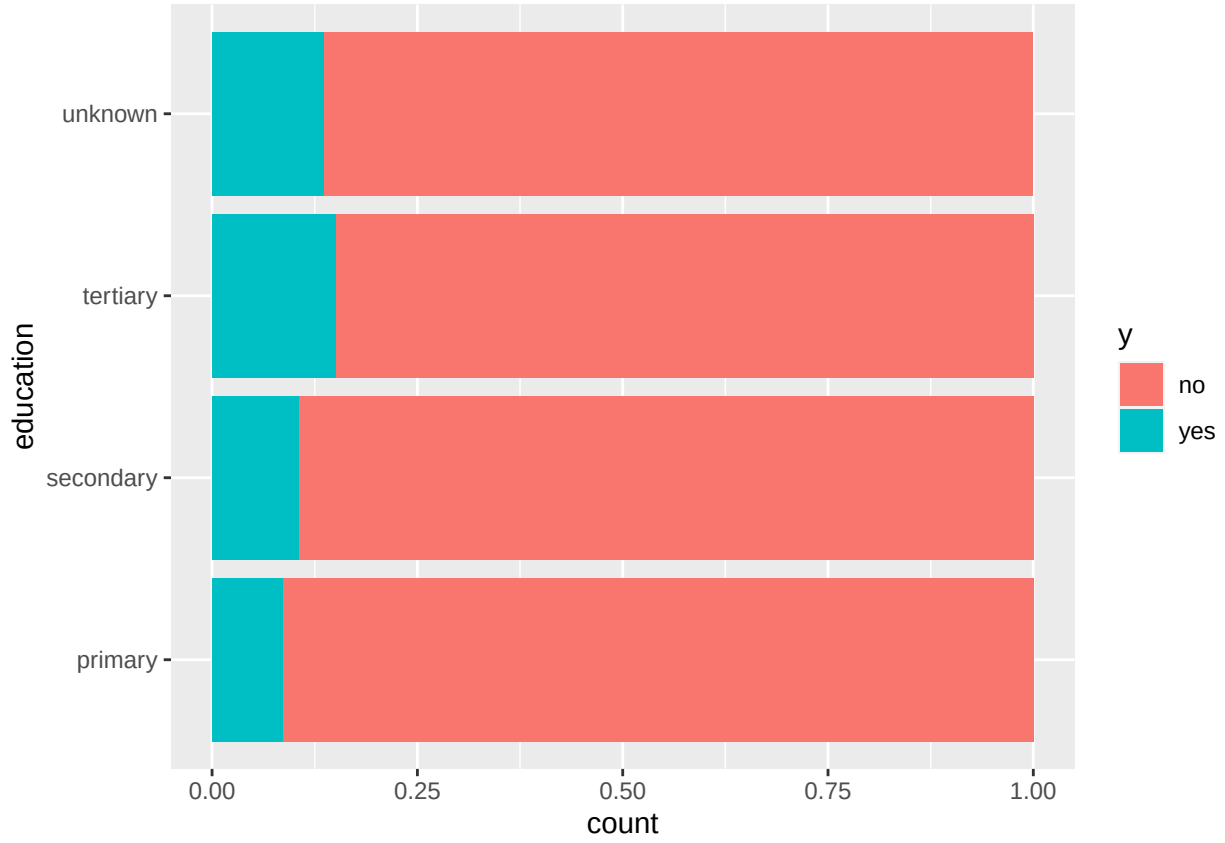
```
##
##    no   yes
## 39922  5289
```

```
##density plot after preprocessing
library(ggplot2)
ggplot(bank,aes(x=age))+geom_density(color='red',fill='lightgreen')
```

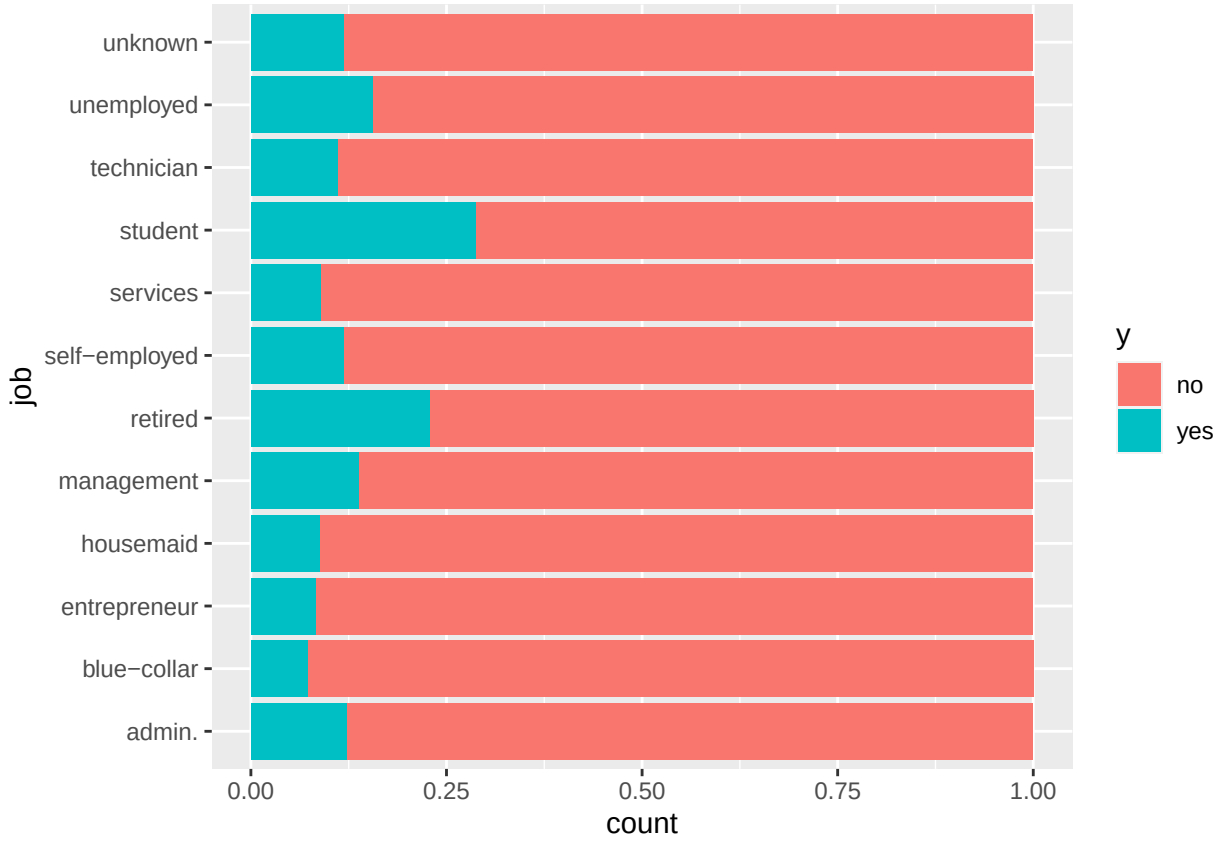

```
ggplot(bank,aes(x=duration))+geom_density(color='red',fill='lightgreen')
```




```
#ggplot for input (education) and output(y)
ggplot(bank, aes(education)) +
  geom_bar(aes(fill = y), position = "fill") +
  coord_flip()
```




```
      #OBSERVED:From above plot we can observed tertiary education peoples are more subscribed
           #a term deposit(yes) compare to other educations.
#ggplot for input(job) and output(y)
ggplot(bank, aes(job)) +
  geom_bar(aes(fill = y), position = "fill") +
  coord_flip()
```




```
        #OBSERVED:from above plot we can observed students are more subscribed a term deposit(yes)
                  #compare to other jobs.
#label encoding features

bank$default_credit =as.numeric(factor(bank$default_credit)) -1
bank$housing_loan =as.numeric(factor(bank$housing_loan)) -1
bank$personal_loan = as.numeric(factor(bank$personal_loan)) -1
bank$y =as.numeric(factor(bank$y)) -1

library(fastDummies)
dummy=bank[c(2,3,4,9,11,16)]
bank1=fastDummies::dummy_cols(dummy,remove_first_dummy = FALSE)
bank1=bank1[c(-1,-2,-3,-4,-5,-6)]
bankData=cbind(bank,bank1)
bankData=bankData[,c(-2,-3,-4,-9,-11,-16)]
View(head(bankData))
dim(bankData)
```

```
## [1] 45211    49
```

```
colnames(bankData)[13]="job_bluecollar"
colnames(bankData)[18]="job_selfempoyed"


#normalization
```

```
library(caret)
preproc=preProcess(bankData[,c(1,3,7)], method=c("range"))

bankData=predict(preproc,bankData)

#split the data train and test
library(caTools)

bankData=bankData[sample(nrow(bankData)),]

split=sample.split(bankData$y,SplitRatio = 0.75)

training_set2<- subset(bankData, split == TRUE)

test_set2<- subset(bankData, split == FALSE)


###################### MACHINE LEARNING ALGORITMS #########################

############## 1.LOGISTIC REGRESSION ###########

# GLM function use sigmoid curve to produce desirable results
# The output of sigmoid function lies in between 0-1
library(caTools)
training=as.data.frame(bankData)
model=glm(y~.,data=training,family = "binomial")
pred=predict(model,training,type="response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
options(pre=-1)
# Confusion matrix and considering the threshold value as 0.5
confusion<-table(pred>0.5,training$y)
confusion
```

```
##
##             0     1
##   FALSE 38821  3406
##   TRUE   1101  1883
```

```
# Model Accuracy
Accuracy_glm<-sum(diag(confusion)/sum(confusion))
Accuracy_glm
```

```
## [1] 0.9003119
```

```
############## 2.DECISION TREES (CART) ################
library(rpart)

fit=rpart(y~.,data=training_set2,method='class')
```

```
predicted=predict(fit,test_set2,type='class')
pred=table(test_set2$y,predicted)

confusionMatrix(pred)
```

```
## Confusion Matrix and Statistics
##
##    predicted
##        0    1
##   0 9655  325
##   1  854  468
##
##                Accuracy : 0.8957
##                  95% CI : (0.8899, 0.9013)
##     No Information Rate : 0.9298
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.389
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9187
##             Specificity : 0.5902
##          Pos Pred Value : 0.9674
##          Neg Pred Value : 0.3540
##              Prevalence : 0.9298
##          Detection Rate : 0.8543
##    Detection Prevalence : 0.8830
##       Balanced Accuracy : 0.7545
##
##        'Positive' Class : 0
##
```

```
accuracy_cart=sum(diag(pred))/sum(pred)
accuracy_cart
```

```
## [1] 0.8956822
```

```
################### 3.NAIVE BAYES #################

#Naive Bayes
library(e1071)

training_set2$y = as.factor(training_set2$y)
test_set2$y =as.factor(test_set2$y)
bankData$y = as.factor(bankData$y)

model = naiveBayes(y~.,training_set2,na.action = na.pass)

predicted_y = predict(model, test_set2)

table <- table(test_set2$y,predicted_y)
table
```

```
##    predicted_y
##        0    1
##   0 9109  871
##   1  708  614
```

```
library(caret)
confusionMatrix(table)
```

```
## Confusion Matrix and Statistics
##
##    predicted_y
##        0    1
##   0 9109  871
##   1  708  614
##
##                Accuracy : 0.8603
##                  95% CI : (0.8538, 0.8666)
##     No Information Rate : 0.8686
##     P-Value [Acc > NIR] : 0.9955
##
##                   Kappa : 0.358
##
##  Mcnemar's Test P-Value : 4.565e-05
##
##             Sensitivity : 0.9279
##             Specificity : 0.4135
##          Pos Pred Value : 0.9127
##          Neg Pred Value : 0.4644
##              Prevalence : 0.8686
##          Detection Rate : 0.8060
##    Detection Prevalence : 0.8830
##       Balanced Accuracy : 0.6707
##
##        'Positive' Class : 0
##
```

```
accuracy_navai=sum(diag(table))/sum(table)
accuracy_navai
```

```
## [1] 0.8602902
```

```
################## 4.K-NEAREST NEIGHBORS (KNN) ##########
library(class)

knn_model <- knn(training_set2,test_set2,training_set2$y, k=3)
tab4 <- table(test_set2$y, knn_model)
tab4
```

```
##    knn_model
##        0    1
##   0 9844  136
##   1  714  608
```

```r
knn_model2 <- knn(training_set2,test_set2,training_set2$y, k=10)
tab <- table(test_set2$y, knn_model)
tab
```

```
##    knn_model
##        0    1
##   0 9844  136
##   1  714  608
```

```r
confusionMatrix(tab4)
```

```
## Confusion Matrix and Statistics
##
##    knn_model
##        0    1
##   0 9844  136
##   1  714  608
##
##                Accuracy : 0.9248
##                  95% CI : (0.9198, 0.9296)
##     No Information Rate : 0.9342
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.5507
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9324
##             Specificity : 0.8172
##          Pos Pred Value : 0.9864
##          Neg Pred Value : 0.4599
##              Prevalence : 0.9342
##          Detection Rate : 0.8710
##    Detection Prevalence : 0.8830
##       Balanced Accuracy : 0.8748
##
##        'Positive' Class : 0
##
```

```r
accuracy_knn=sum(diag(tab4))/sum(tab4)
accuracy_knn
```

```
## [1] 0.9247921
```

```r
#################### 5.SUPPORT VECTOR MACHINE (SVM) ###############

library(e1071)

train.svm=svm(y~.,training_set2,kernel="polynomial",cost=0.01,scale=TRUE)

#  This is where we determine the model accuracy.  predict() applies the model to the test data.
```

```r
test.svm=predict(train.svm,test_set2)

#table() shows the contingency table for the outcome of the prediction.  Each cell in a contingency tab
pred=table(test_set2$y,test.svm)
#confusion matrix
confusionMatrix(pred)
```

```
## Confusion Matrix and Statistics
##
##    test.svm
##        0    1
##   0 9886   94
##   1 1095  227
##
##                Accuracy : 0.8948
##                  95% CI : (0.889, 0.9004)
##     No Information Rate : 0.9716
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2417
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9003
##             Specificity : 0.7072
##          Pos Pred Value : 0.9906
##          Neg Pred Value : 0.1717
##              Prevalence : 0.9716
##          Detection Rate : 0.8747
##    Detection Prevalence : 0.8830
##       Balanced Accuracy : 0.8037
##
##        'Positive' Class : 0
##
```

```r
#model accuracy
accuracy_svm=sum(diag(pred)/sum(pred))
accuracy_svm
```

```
## [1] 0.8947974
```

```r
##################### 6.RANDOM FOREST #################
# Building a random forest model on training data
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
model=randomForest(y~.,data=training_set2)

predict <- predict(model,test_set2)

pred=table(test_set2$y,predict)
#confusion matrix
confusionMatrix(pred)
```

```
## Confusion Matrix and Statistics
##
##    predict
##        0    1
##   0 9751  229
##   1  875  447
##
##                Accuracy : 0.9023
##                  95% CI : (0.8967, 0.9077)
##     No Information Rate : 0.9402
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.4
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9177
##             Specificity : 0.6612
##          Pos Pred Value : 0.9771
##          Neg Pred Value : 0.3381
##              Prevalence : 0.9402
##          Detection Rate : 0.8628
##    Detection Prevalence : 0.8830
##       Balanced Accuracy : 0.7894
##
##        'Positive' Class : 0
##
```

```
#model accuracy
accuracy_raf=sum(diag(pred)/sum(pred))
accuracy_raf
```

```
## [1] 0.9023182
```

```
######## MACHINE LEARNING MODELS ACCURACY ########

Accuracy_glm
```

```
## [1] 0.9003119
```

```
accuracy_cart
```

```
## [1] 0.8956822
```

```
accuracy_navai
```

```
## [1] 0.8602902
```

```
accuracy_knn
```

```
## [1] 0.9247921
```

```
accuracy_svm
```

```
## [1] 0.8947974
```

```
accuracy_raf
```

```
## [1] 0.9023182
```

```
################ OBERVATIONS #################

#### 1.ggplots for input and output variables
 #JOB:"student" are more subscribed(25%) compare to other jobs
 #EDUCATION:"tertiary" are more subscribed(16%) compare to other education

#### 2.machine learning models
    #on appling knn algorithm the accuracy is 92.30% hence by comparing with
     #other algorithms knn is best algorithm for this data set.
        #deploy final model of knn model.
```