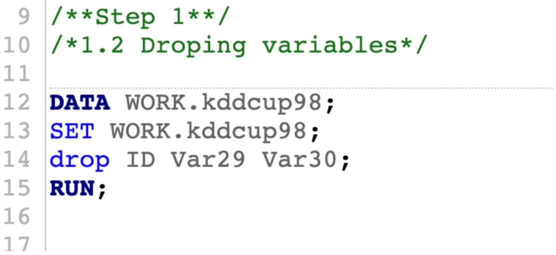
Amin Baabol
Partner: Mohamed Ahmed
INFS 762
Project 1
September 30th, 2020

Task 1: Classification (with TARGET_B as the dependent variable)

Step 1

1.1: Importing
1.2: Code



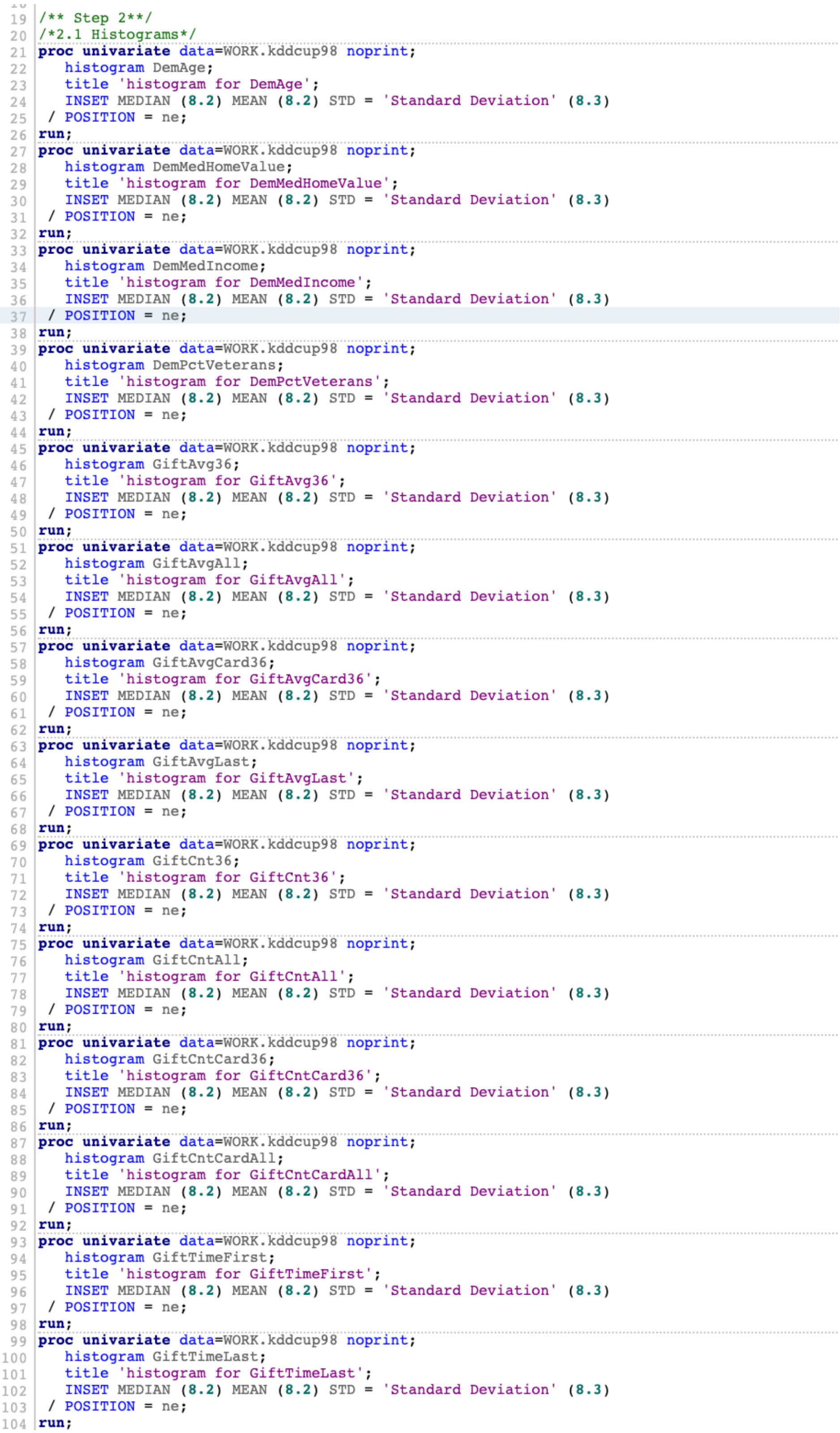
```
/**Step 1**/
/*1.2 Droping variables*/

DATA WORK.kddcup98;
SET WORK.kddcup98;
drop ID Var29 Var30;
RUN;


```

Step 2: Data exploration

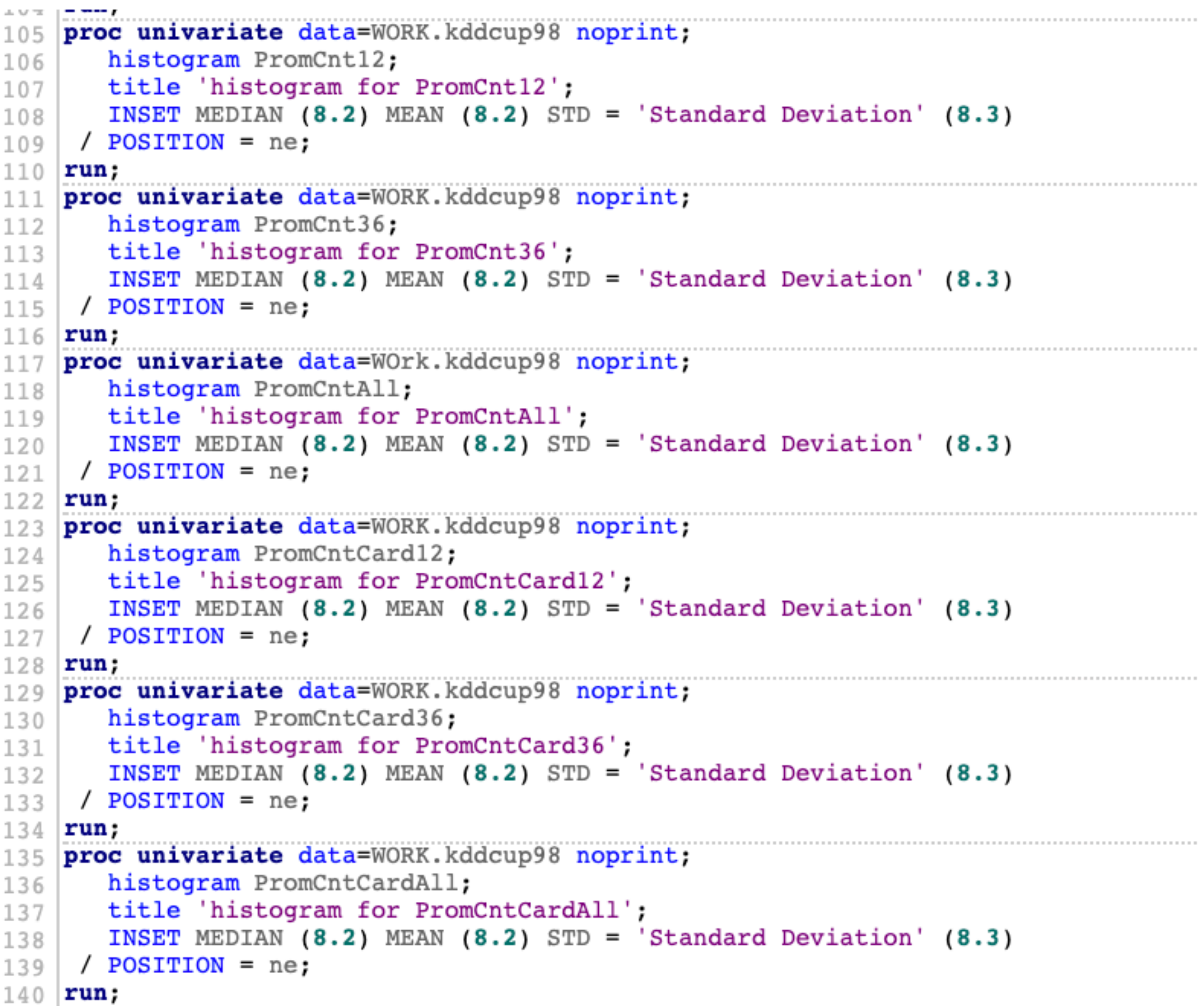
2.1: Histograms
Variable list:

- DemAge
- DemMedHomeValue
- DemMedIncome
- DemPctVeterans
- GiftAvg36
- GiftAvgAll
- GiftAvgCard36
- GiftAvgLast
- GiftCnt36
- GiftCntAll
- GiftCntCard36
- GiftCntCardAll
- GiftTimeFirst
- GiftTimeLast
- PromCnt12
- PromCnt36
- PromCntAll
- PromCntCard12
- PromCntCard36
- PromCntCardAll

Code

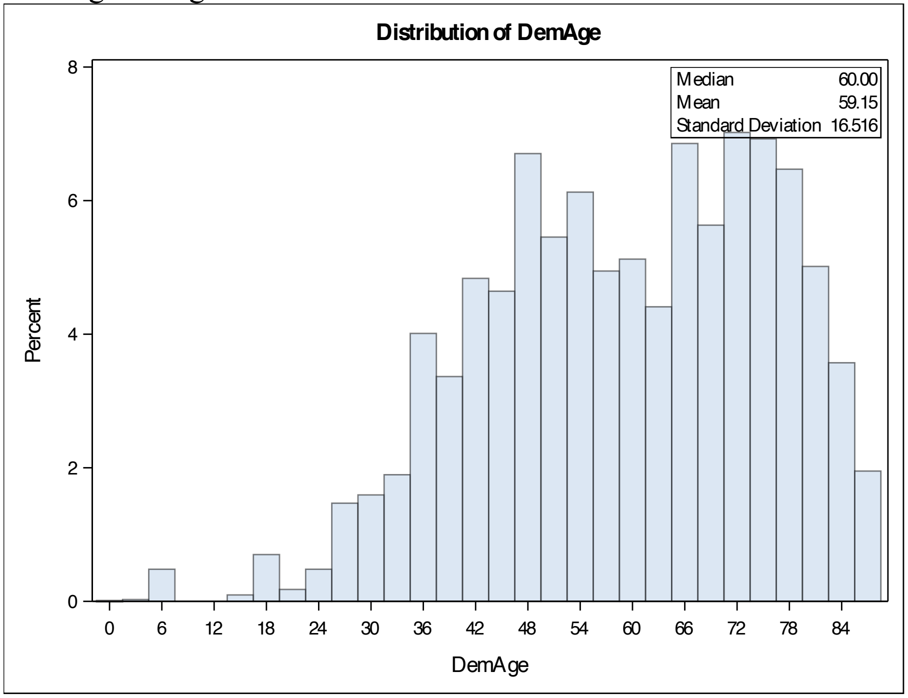


```
/** Step 2**/
/*2.1 Histograms*/
proc univariate data=WORK.kddcup98 noprint;
   histogram DemAge;
   title 'histogram for DemAge';
   INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
   histogram DemMedHomeValue;
   title 'histogram for DemMedHomeValue';
   INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
   histogram DemMedIncome;
   title 'histogram for DemMedIncome';
   INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
   histogram DemPctVeterans;
   title 'histogram for DemPctVeterans';
   INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
   histogram GiftAvg36;
   title 'histogram for GiftAvg36';
   INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
   histogram GiftAvgAll;
   title 'histogram for GiftAvgAll';
   INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
   histogram GiftAvgCard36;
   title 'histogram for GiftAvgCard36';
   INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
   histogram GiftAvgLast;
   title 'histogram for GiftAvgLast';
   INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
   histogram GiftCnt36;
   title 'histogram for GiftCnt36';
   INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
   histogram GiftCntAll;
   title 'histogram for GiftCntAll';
   INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
   histogram GiftCntCard36;
   title 'histogram for GiftCntCard36';
   INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
   histogram GiftCntCardAll;
   title 'histogram for GiftCntCardAll';
   INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
   histogram GiftTimeFirst;
   title 'histogram for GiftTimeFirst';
   INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
   histogram GiftTimeLast;
   title 'histogram for GiftTimeLast';
   INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
```

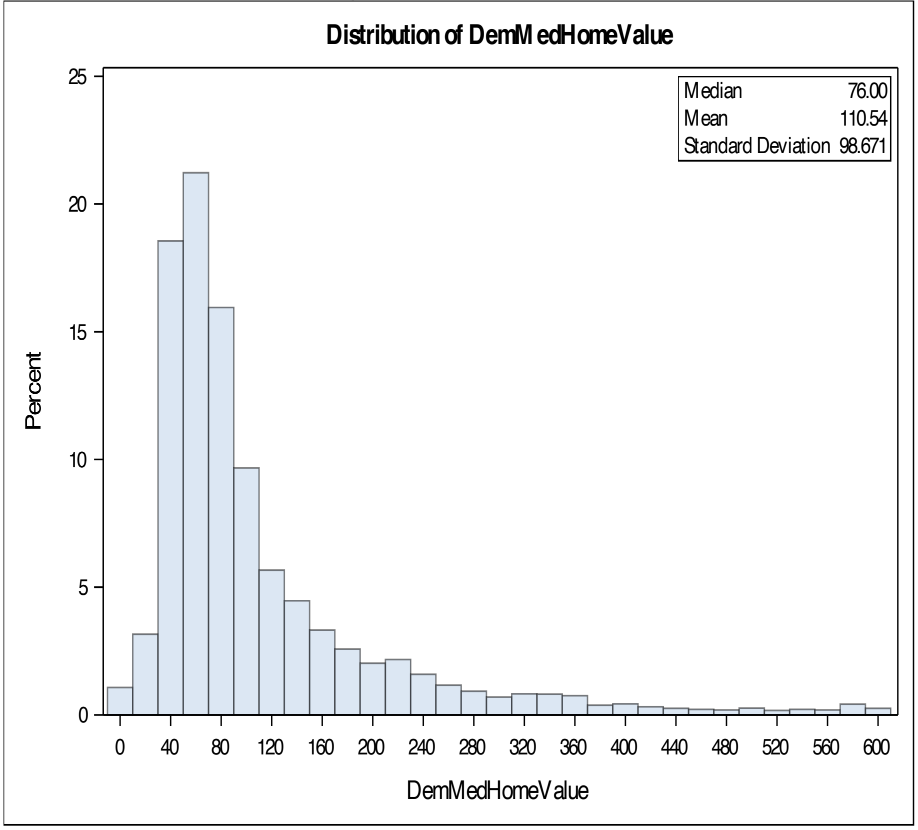


```
run;
proc univariate data=WORK.kddcup98 noprint;
    histogram PromCnt12;
    title 'histogram for PromCnt12';
    INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
    histogram PromCnt36;
    title 'histogram for PromCnt36';
    INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WOrk.kddcup98 noprint;
    histogram PromCntAll;
    title 'histogram for PromCntAll';
    INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
    histogram PromCntCard12;
    title 'histogram for PromCntCard12';
    INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
    histogram PromCntCard36;
    title 'histogram for PromCntCard36';
    INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
proc univariate data=WORK.kddcup98 noprint;
    histogram PromCntCardAll;
    title 'histogram for PromCntCardAll';
    INSET MEDIAN (8.2) MEAN (8.2) STD = 'Standard Deviation' (8.3)
 / POSITION = ne;
run;
```

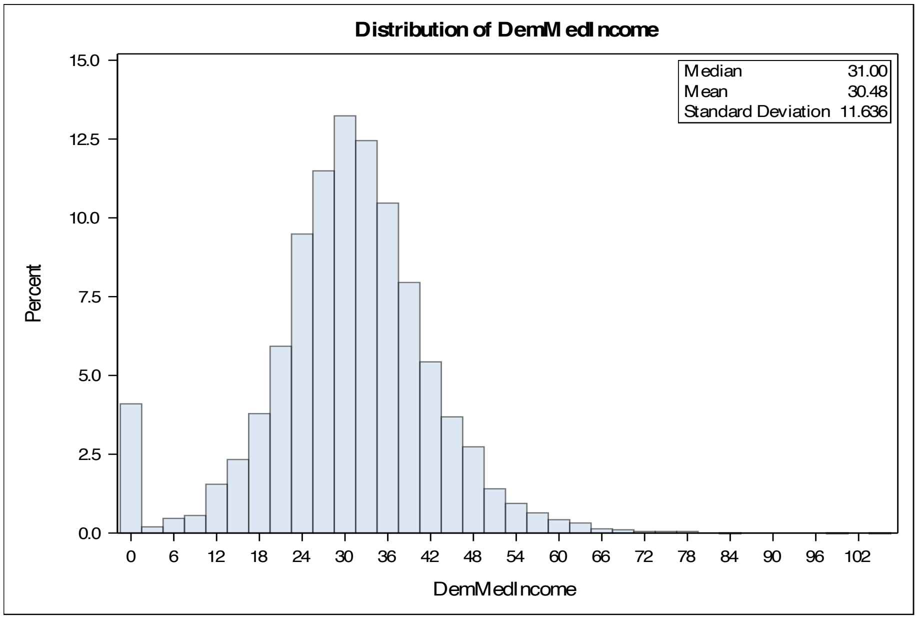
DemAge histogram



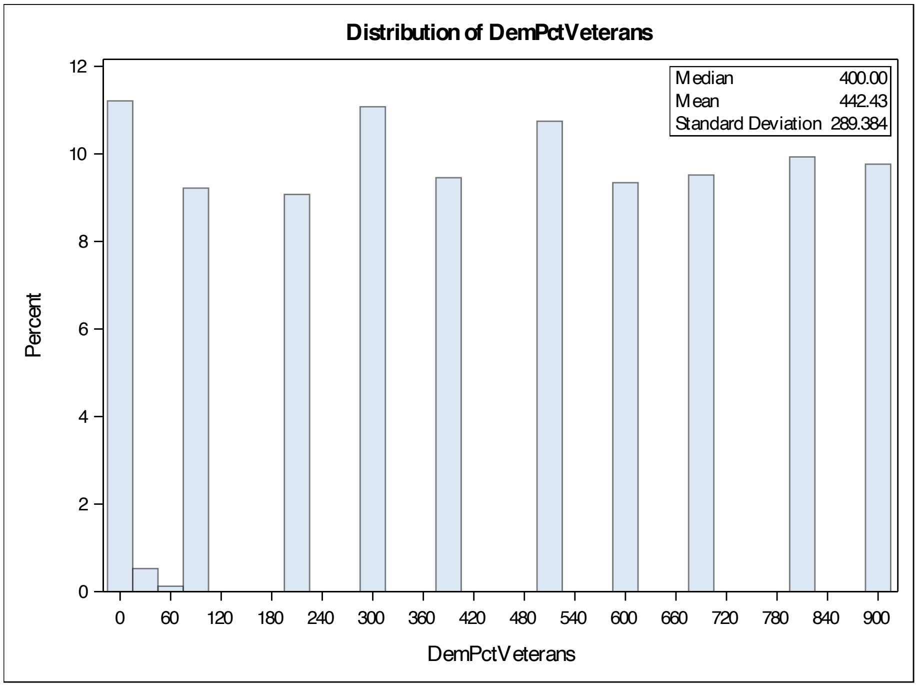
DemMedHomeValue histogram



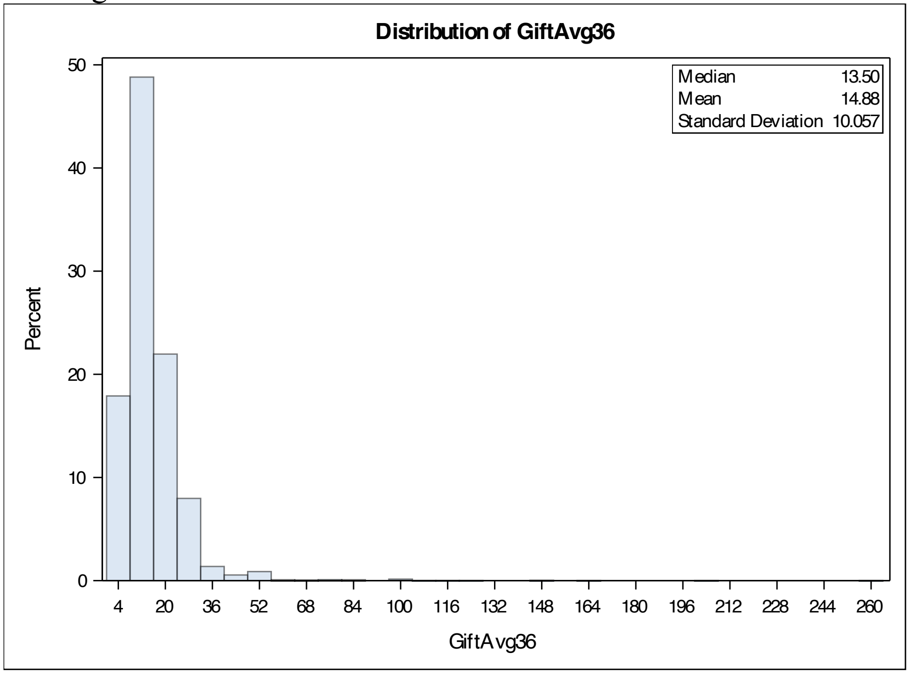
DemMedIncome



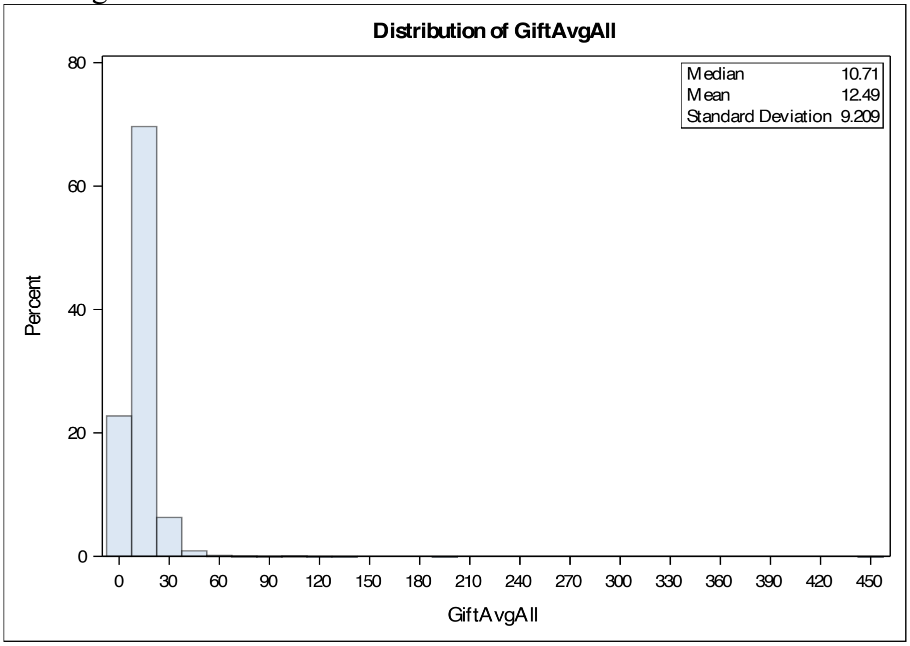
DemPctVeterans



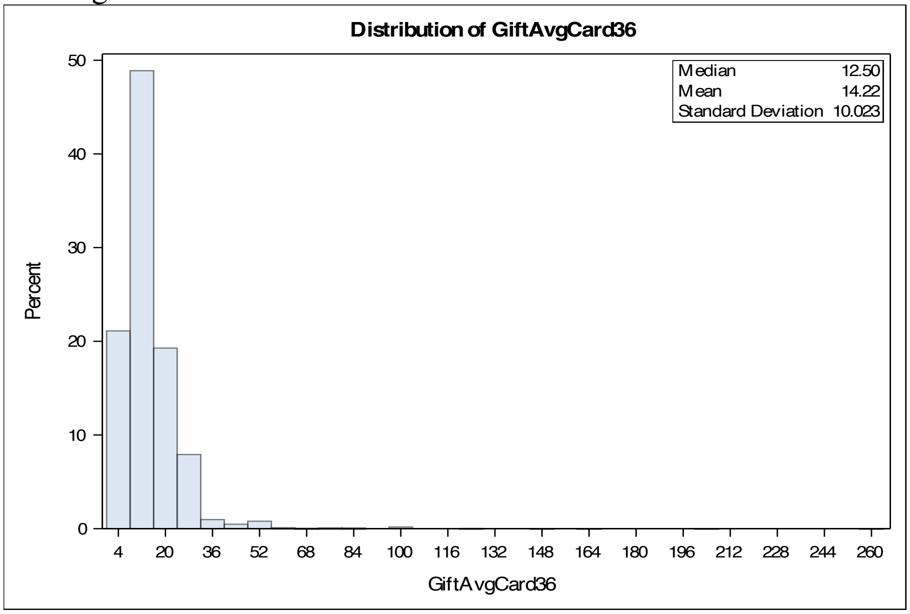
GiftAvg36



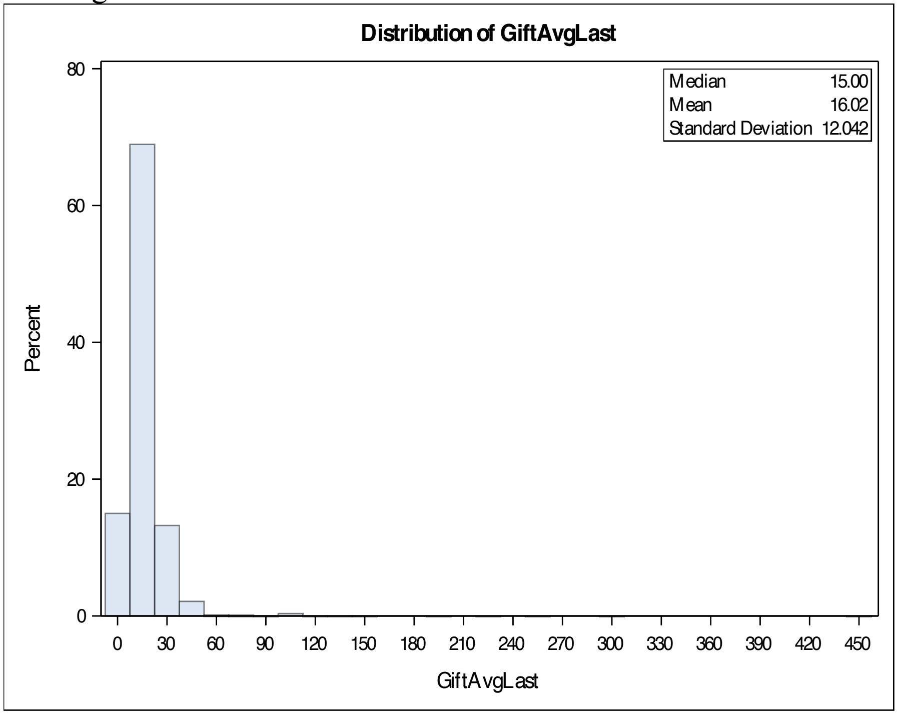
GiftAvgAll



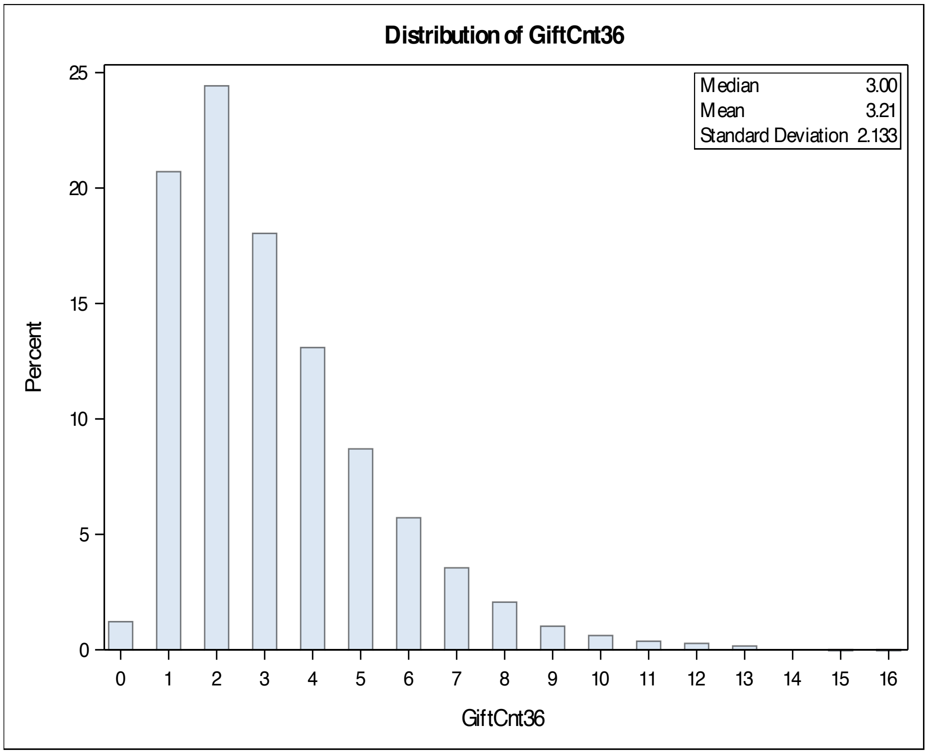
GiftAvgCard36



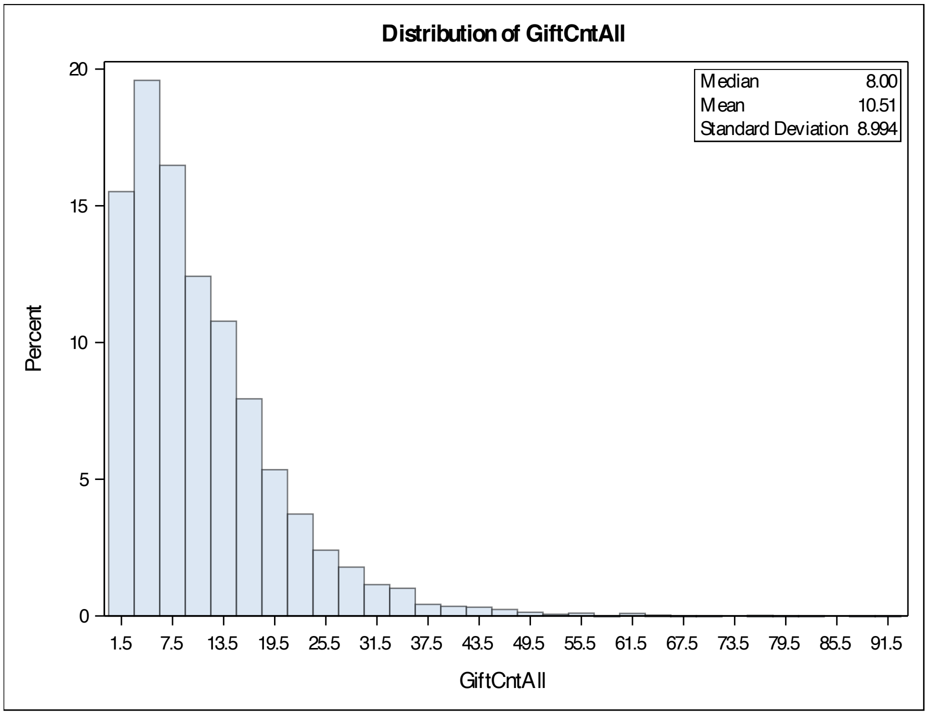
GiftAvgLast



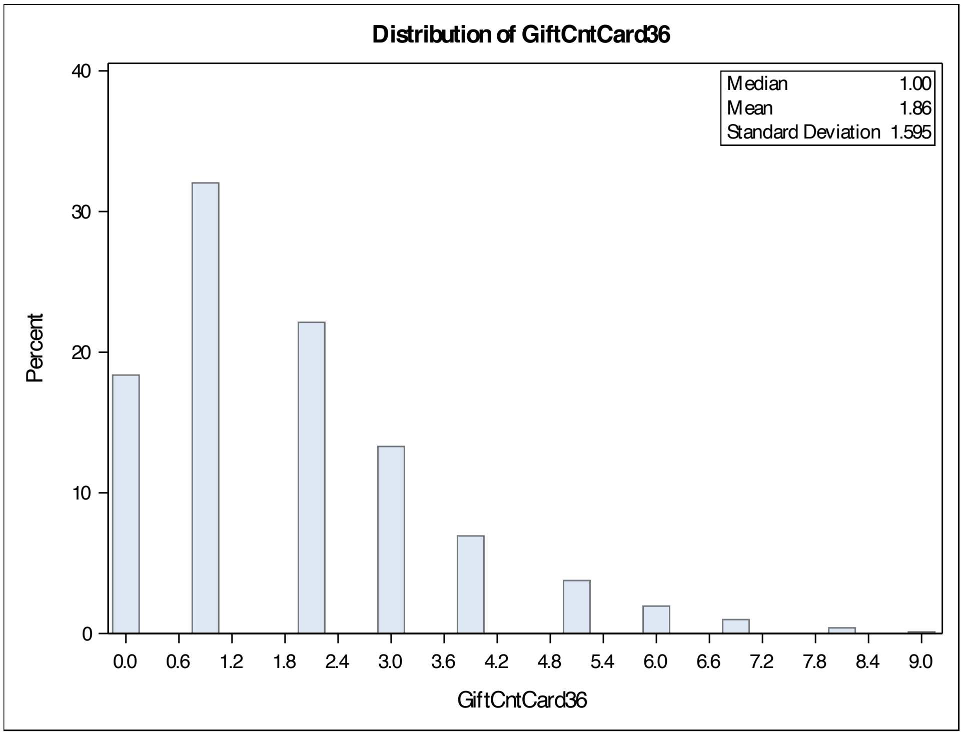
GiftCnt36



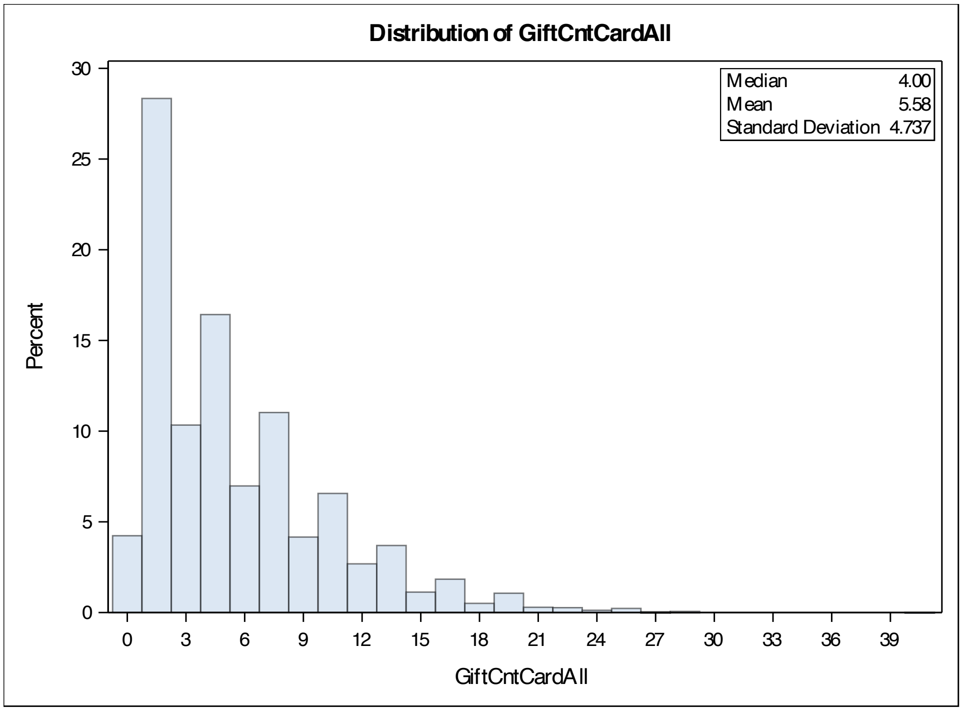
GiftCntAll



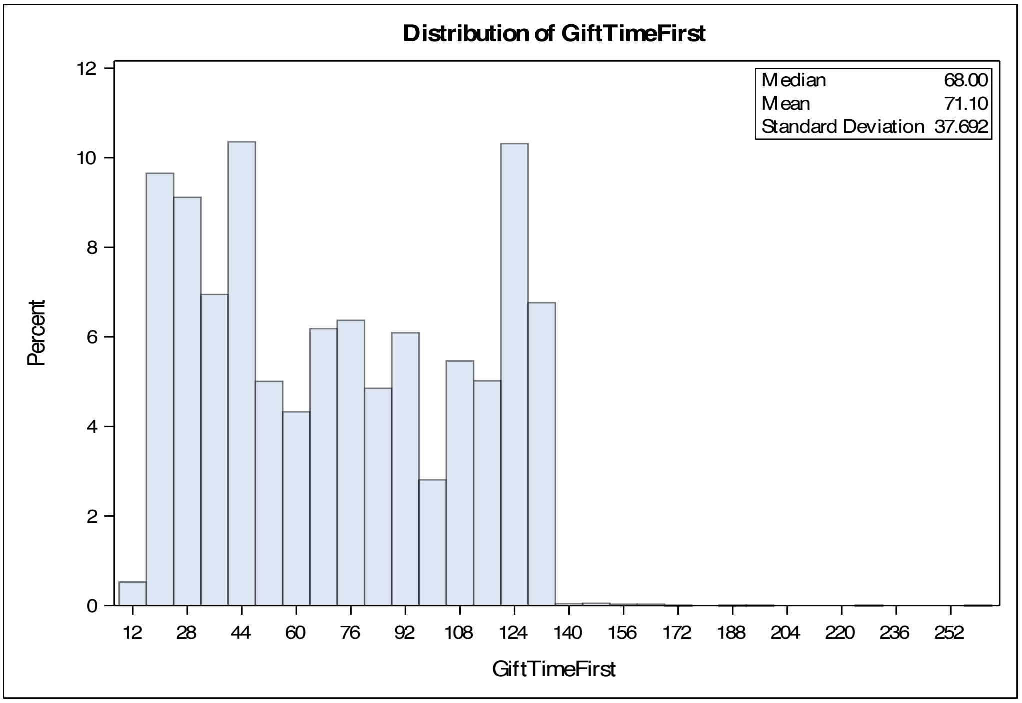
GiftCntCard36



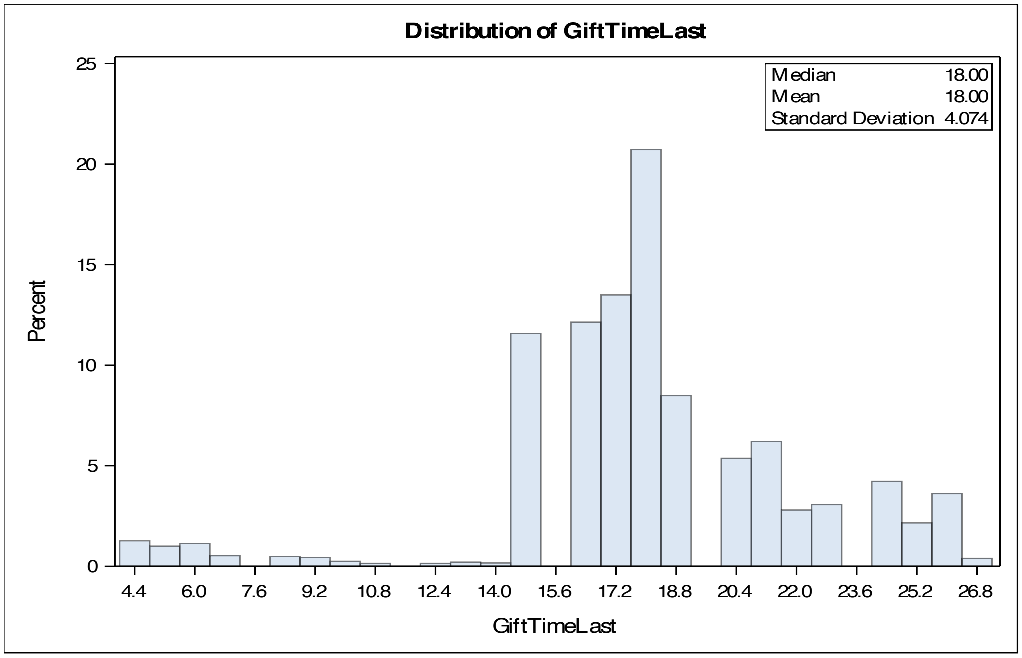
GiftCntCardAll



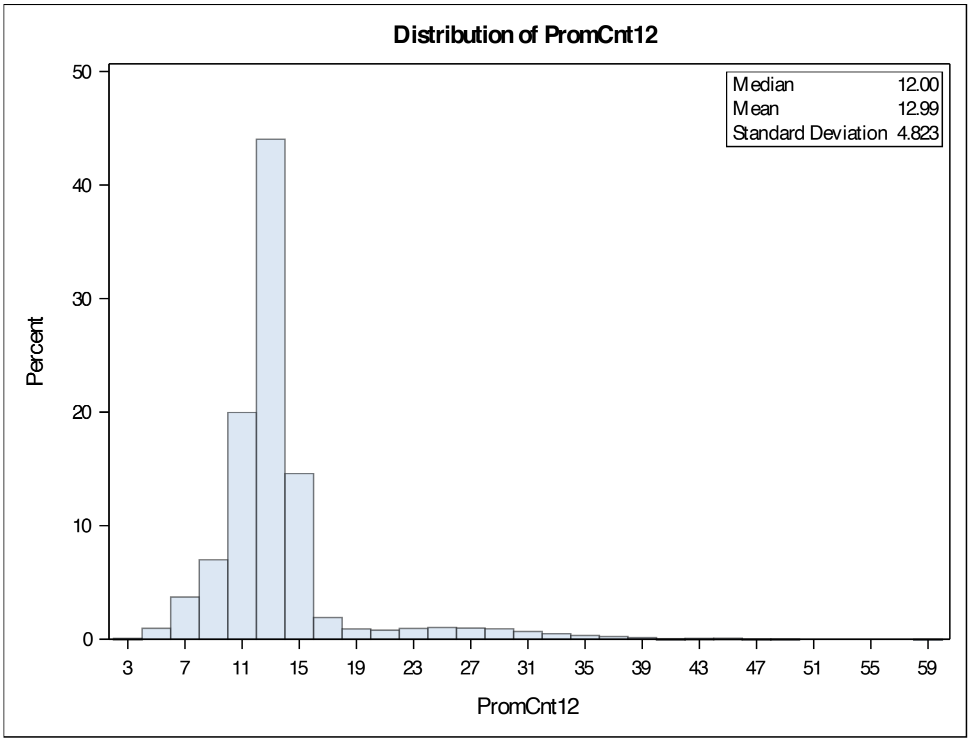
GiftTimeFirst



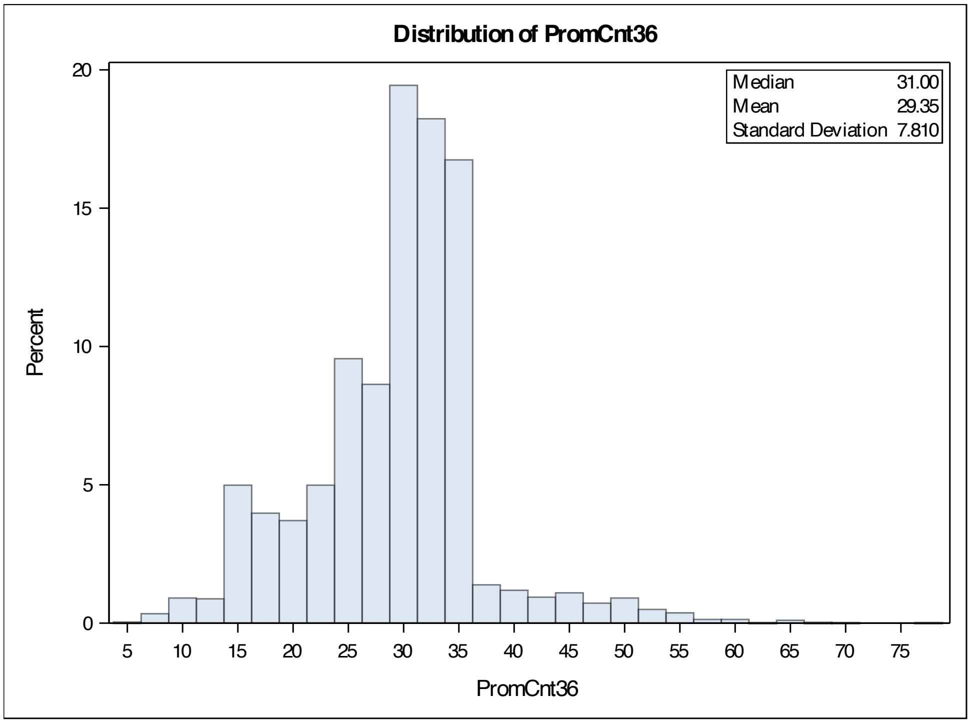
GiftTimeLast



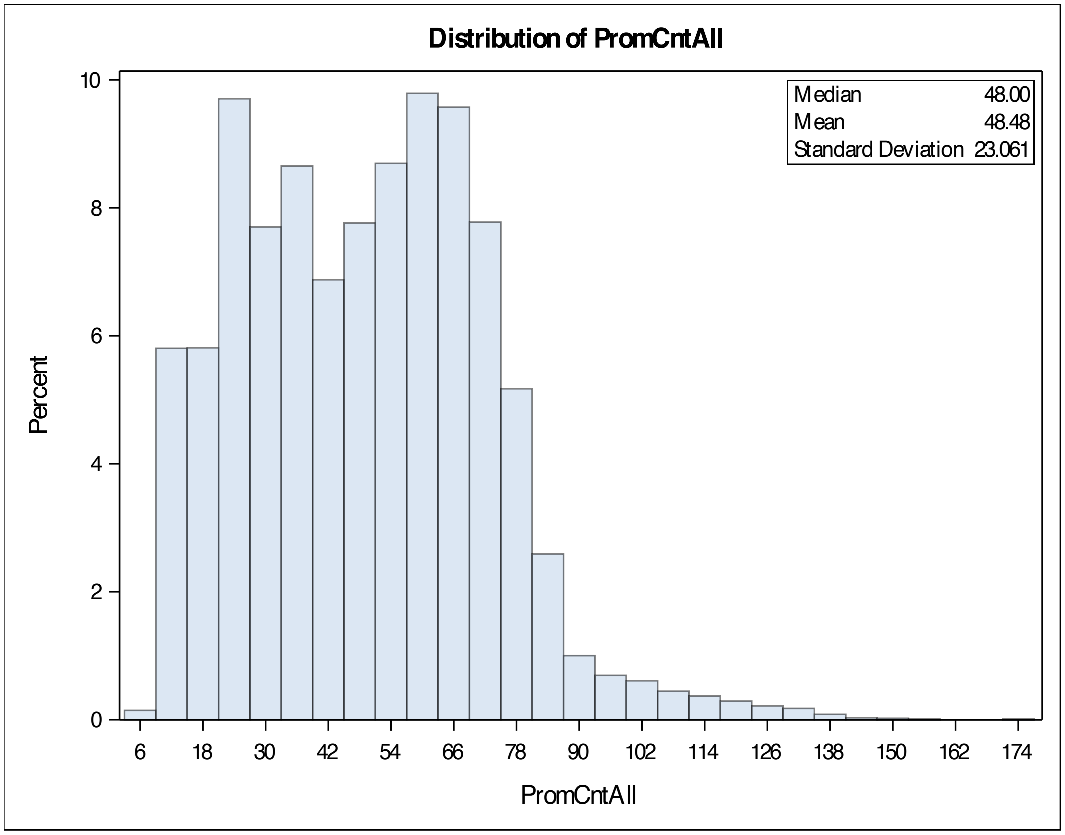
PromCnt12



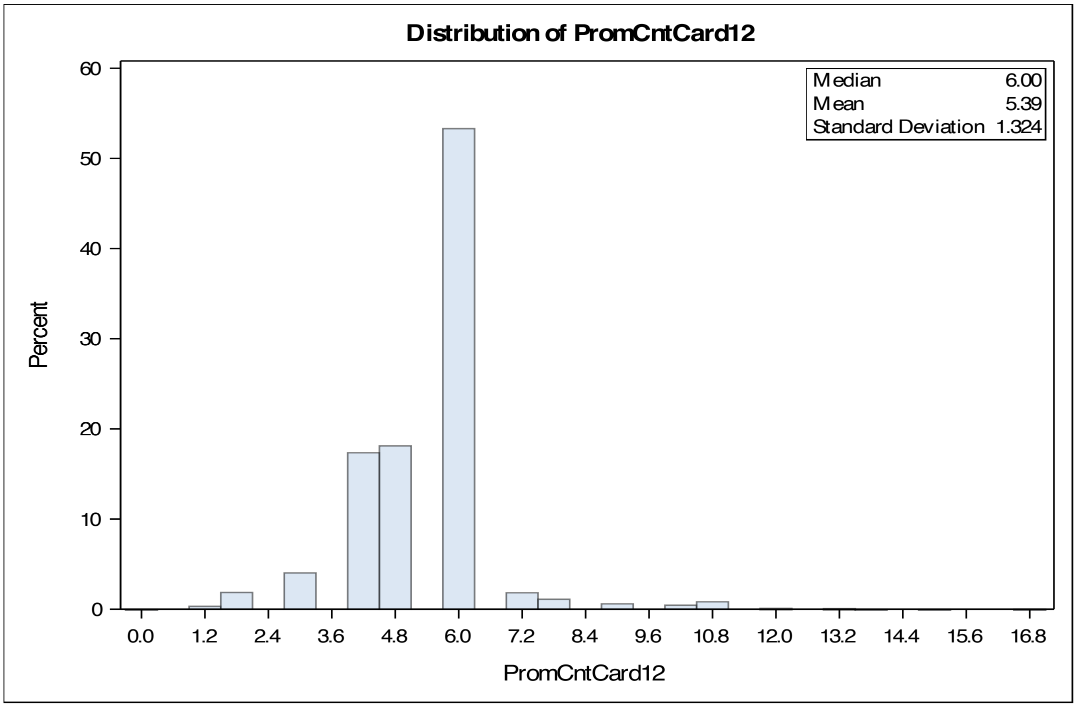
PromCnt36



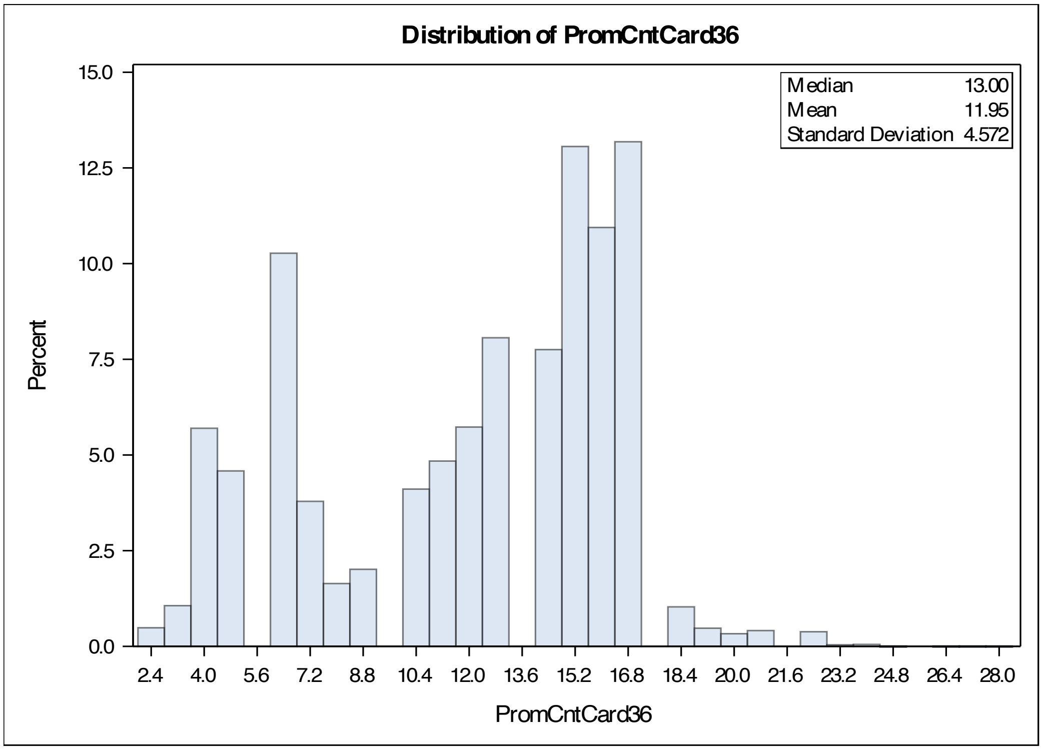
PromCntAll



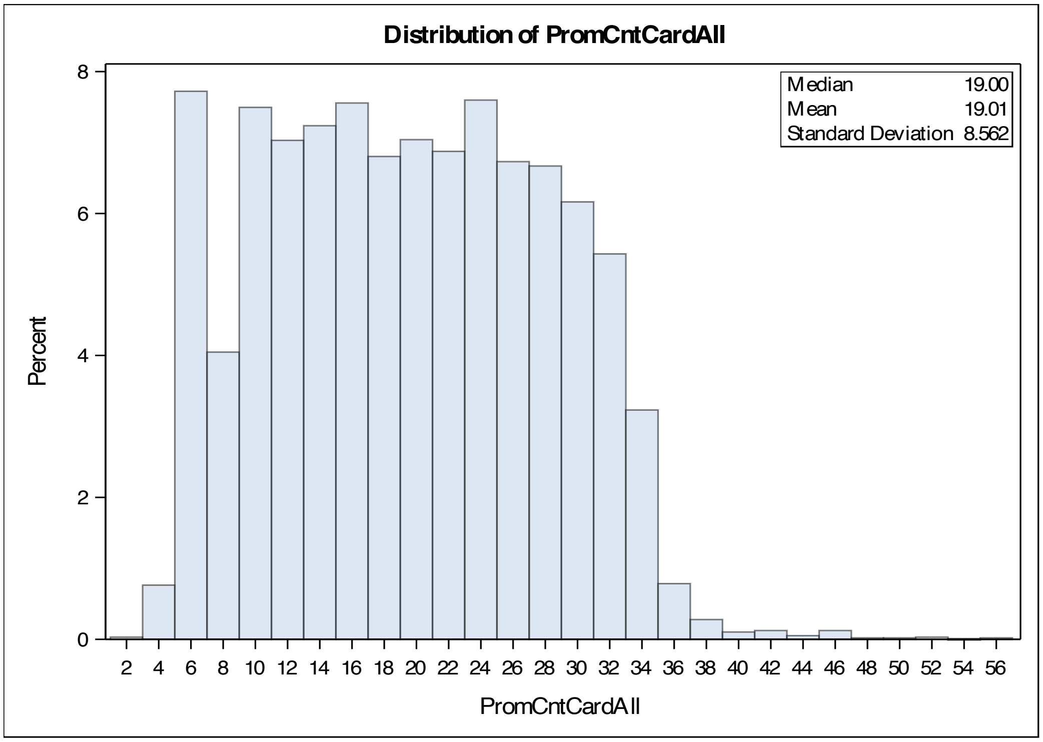
PromCntCard12



PromCntCad36



PromCntCardAll



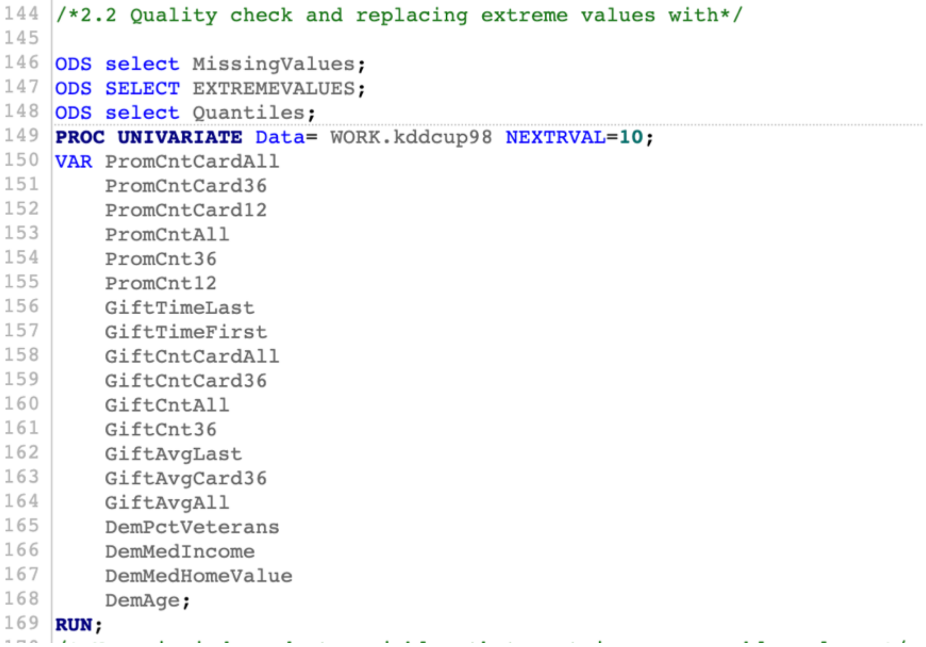
2.2: Quality check

We first checked the histogram distribution(mean &median) of each numeric variable for anomalies that make no sense, then we proceeded by using the 99 percentile and 1 percentile as our extreme bounds, any value above or below those percentiles will be considered extreme value except for variables with good normal distribution. we found that a lot of the numeric independent variables contain extreme values which caused the distribution of the data to have skewness.

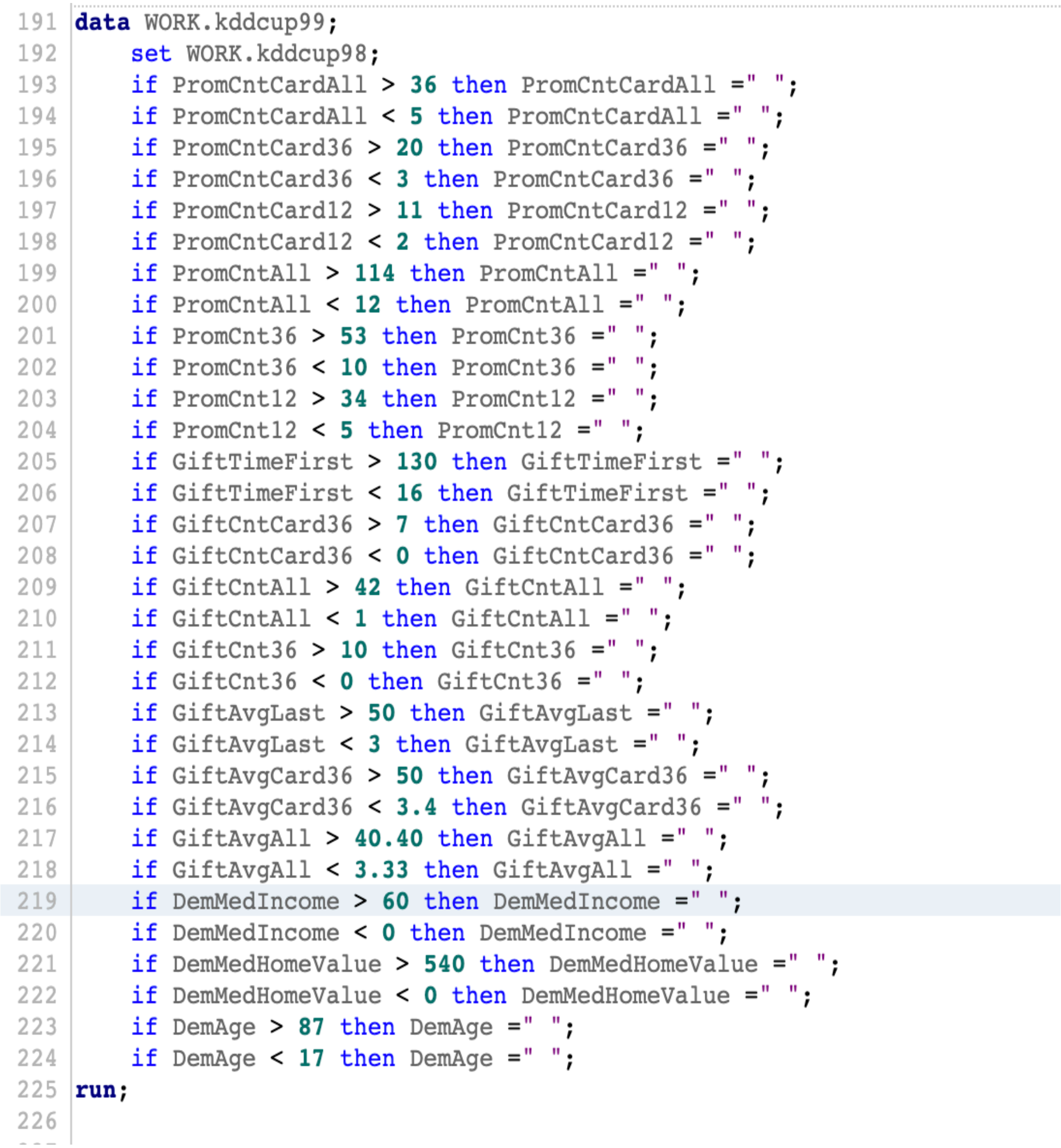
List of variables with unreasonable values

- PromCntCardAll (36,5)
- PromCntCard36 (20,3)
- PromCntCard12 (11,2)
- PromCntAll (114,12)
- PromCnt36 (53,10)
- PromCnt12 (34,5)
- GiftTimeFirst (130,16)
- GiftCntCard36 (7,0)
- GiftCntAll (42,1)
- GiftCnt36 (10,0)
- GiftAvgLast (50,3)
- GiftAvgCard3 (50,3.4)
- GiftAvgAll (40.40,3.33)
- DemMedIncome (60,0)
- DemMedHomeValue (540,0)
- DemAge (87,17)

Code: identify and replacing unreasonable values with missing



```
/*2.2 Quality check and replacing extreme values with*/

ODS select MissingValues;
ODS SELECT EXTREMEVALUES;
ODS select Quantiles;
PROC UNIVARIATE Data= WORK.kddcup98 NEXTRVAL=10;
VAR PromCntCardAll
    PromCntCard36
    PromCntCard12
    PromCntAll
    PromCnt36
    PromCnt12
    GiftTimeLast
    GiftTimeFirst
    GiftCntCardAll
    GiftCntCard36
    GiftCntAll
    GiftCnt36
    GiftAvgLast
    GiftAvgCard36
    GiftAvgAll
    DemPctVeterans
    DemMedIncome
    DemMedHomeValue
    DemAge;
RUN;
```



```
data WORK.kddcup99;
    set WORK.kddcup98;
    if PromCntCardAll > 36 then PromCntCardAll =" ";
    if PromCntCardAll < 5 then PromCntCardAll =" ";
    if PromCntCard36 > 20 then PromCntCard36 =" ";
    if PromCntCard36 < 3 then PromCntCard36 =" ";
    if PromCntCard12 > 11 then PromCntCard12 =" ";
    if PromCntCard12 < 2 then PromCntCard12 =" ";
    if PromCntAll > 114 then PromCntAll =" ";
    if PromCntAll < 12 then PromCntAll =" ";
    if PromCnt36 > 53 then PromCnt36 =" ";
    if PromCnt36 < 10 then PromCnt36 =" ";
    if PromCnt12 > 34 then PromCnt12 =" ";
    if PromCnt12 < 5 then PromCnt12 =" ";
    if GiftTimeFirst > 130 then GiftTimeFirst =" ";
    if GiftTimeFirst < 16 then GiftTimeFirst =" ";
    if GiftCntCard36 > 7 then GiftCntCard36 =" ";
    if GiftCntCard36 < 0 then GiftCntCard36 =" ";
    if GiftCntAll > 42 then GiftCntAll =" ";
    if GiftCntAll < 1 then GiftCntAll =" ";
    if GiftCnt36 > 10 then GiftCnt36 =" ";
    if GiftCnt36 < 0 then GiftCnt36 =" ";
    if GiftAvgLast > 50 then GiftAvgLast =" ";
    if GiftAvgLast < 3 then GiftAvgLast =" ";
    if GiftAvgCard36 > 50 then GiftAvgCard36 =" ";
    if GiftAvgCard36 < 3.4 then GiftAvgCard36 =" ";
    if GiftAvgAll > 40.40 then GiftAvgAll =" ";
    if GiftAvgAll < 3.33 then GiftAvgAll =" ";
    if DemMedIncome > 60 then DemMedIncome =" ";
    if DemMedIncome < 0 then DemMedIncome =" ";
    if DemMedHomeValue > 540 then DemMedHomeValue =" ";
    if DemMedHomeValue < 0 then DemMedHomeValue =" ";
    if DemAge > 87 then DemAge =" ";
    if DemAge < 17 then DemAge =" ";
run;
```

2.3- Numeric independent variables with missing values

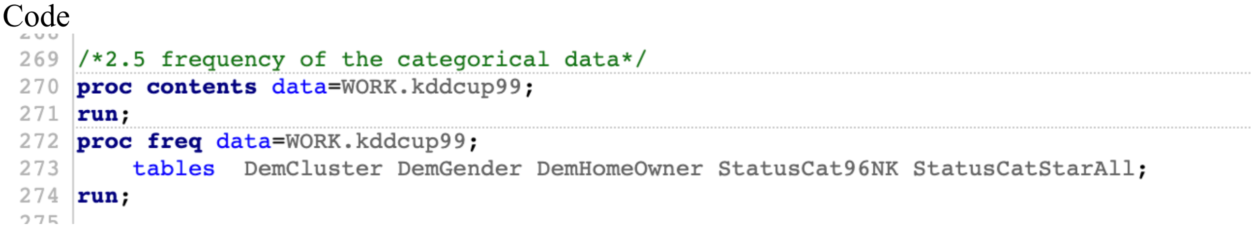
- GiftCnt36
- GiftCntAll
- GiftCntCard36
- GiftAvgLast
- GiftAvgAll
- GiftAvgCard36
- GiftTimeFirst
- PromCnt12
- PromCnt36
- PromCntAll
- PromCntCard12
- PromCntCard36
- PromCntCardAll
- DemAge
- DemMedHomeValue
- DemMedIncome

2.4- Right-skewed numeric independent variables

- DemMedHomeValue
- GiftAvg36
- GiftAvgAll
- GiftAvgCard36
- GiftAvgLast
- GiftCnt36
- GiftCntAll
- GiftCntCard36
- GiftCntCardAll
- GiftTimeFirst
- PromCnt12
- PromCntAll
- PromCntCardAll

2.5- Frequency table for the categorical independent variables

Code



```
/*2.5 frequency of the categorical data*/
proc contents data=WORK.kddcup99;
run;
proc freq data=WORK.kddcup99;
    tables  DemCluster DemGender DemHomeOwner StatusCat96NK StatusCatStarAll;
run;
```

DemCluster	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	240	2.48	240	2.48
1	121	1.25	361	3.73
2	191	1.97	552	5.70
3	153	1.58	705	7.28
4	51	0.53	756	7.81
5	95	0.98	851	8.79
6	53	0.55	904	9.33
7	78	0.81	982	10.14
8	182	1.88	1164	12.02
9	70	0.72	1234	12.74
10	175	1.81	1409	14.55
11	236	2.44	1645	16.98
12	323	3.33	1968	20.32
13	309	3.19	2277	23.51
14	248	2.56	2525	26.07
15	108	1.12	2633	27.18
16	201	2.08	2834	29.26
17	178	1.84	3012	31.10
18	321	3.31	3333	34.41
19	50	0.52	3383	34.93
20	171	1.77	3554	36.69
21	165	1.70	3719	38.40
22	125	1.29	3844	39.69
23	131	1.35	3975	41.04
24	401	4.14	4376	45.18
25	135	1.39	4511	46.57
26	100	1.03	4611	47.60
27	331	3.42	4942	51.02
28	194	2.00	5136	53.02
29	73	0.75	5209	53.78
30	262	2.70	5471	56.48
31	125	1.29	5596	57.77
32	72	0.74	5668	58.52
33	52	0.54	5720	59.05
34	132	1.36	5852	60.42
35	384	3.96	6236	64.38
36	401	4.14	6637	68.52
37	99	1.02	6736	69.54
38	118	1.22	6854	70.76
39	242	2.50	7096	73.26
40	432	4.46	7528	77.72
41	197	2.03	7725	79.75
42	140	1.45	7865	81.20

DemCluster	Frequency	Percent	Cumulative Frequency	Cumulative Percent
43	227	2.34	8092	83.54
44	185	1.91	8277	85.45
45	228	2.35	8505	87.81
46	196	2.02	8701	89.83
47	86	0.89	8787	90.72
48	96	0.99	8883	91.71
49	323	3.33	9206	95.04
50	70	0.72	9276	95.77
51	220	2.27	9496	98.04
52	32	0.33	9528	98.37
53	158	1.63	9686	100.00

DemGender	Frequency	Percent	Cumulative Frequency	Cumulative Percent
F	5223	53.92	5223	53.92
M	3925	40.52	9148	94.45
U	538	5.55	9686	100.00

DemHomeOwner	Frequency	Percent	Cumulative Frequency	Cumulative Percent
H	5377	55.51	5377	55.51
U	4309	44.49	9686	100.00

StatusCat96NK	Frequency	Percent	Cumulative Frequency	Cumulative Percent
A	5826	63.94	5826	63.94
E	227	2.49	6053	66.43
F	660	7.24	6713	73.67
L	34	0.37	6747	74.05
S	2365	25.95	9112	100.00
Frequency Missing = 574				

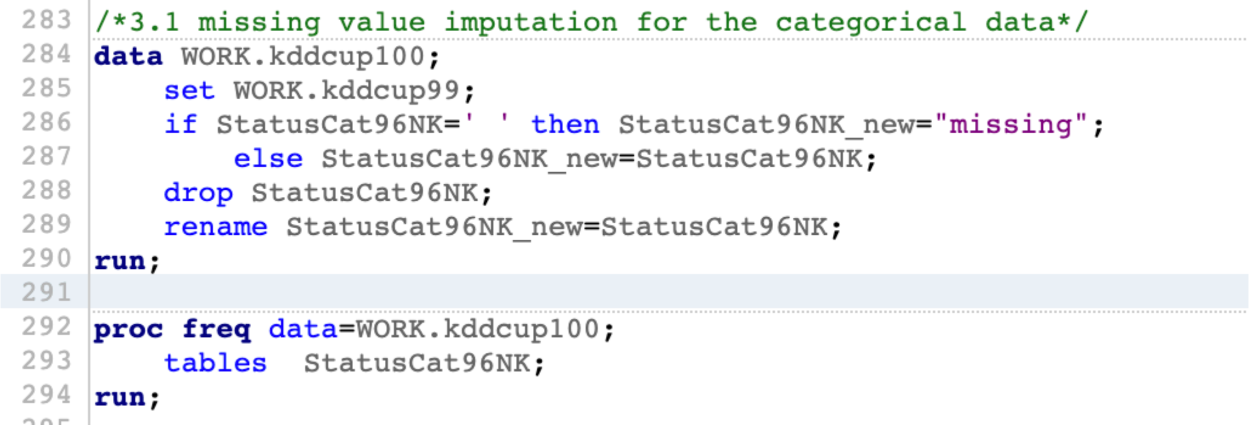
StatusCatStarAll	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	4450	45.94	4450	45.94
1	5236	54.06	9686	100.00

2.6- Categorical independent variables with missing values

- StatusCat96NK

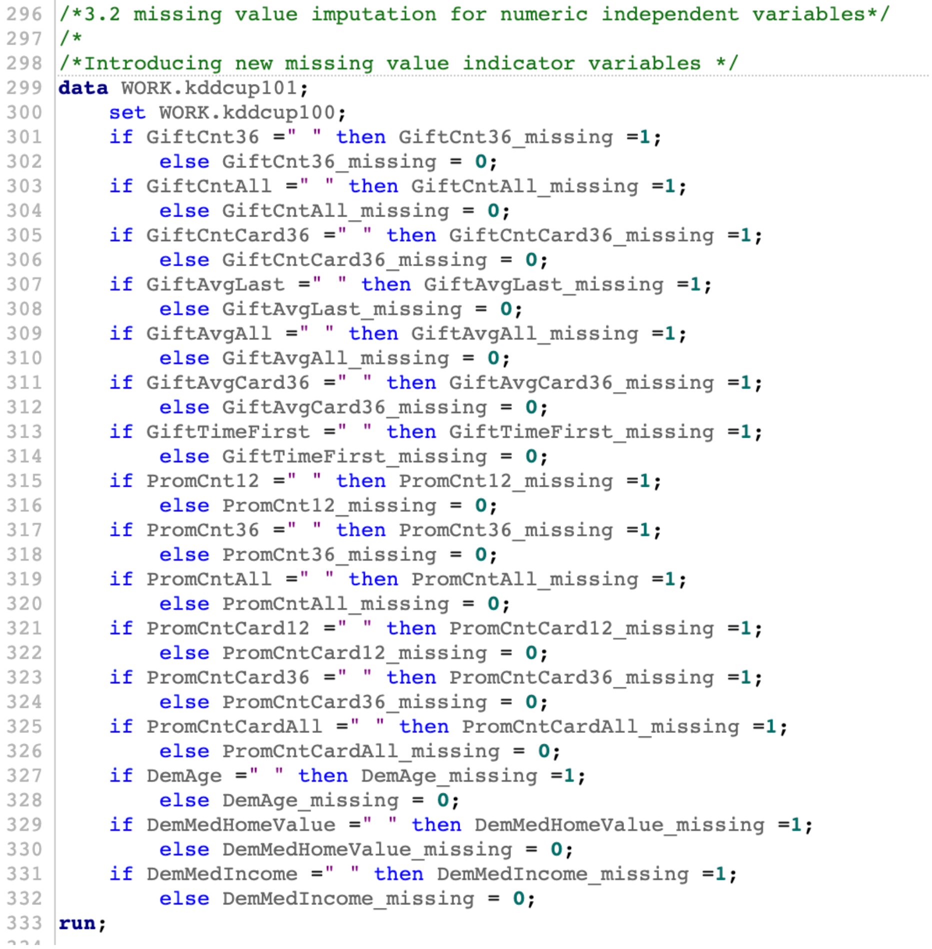
Step 3: Variable Transformation

3.1- missing value imputation for the categorical independent variables

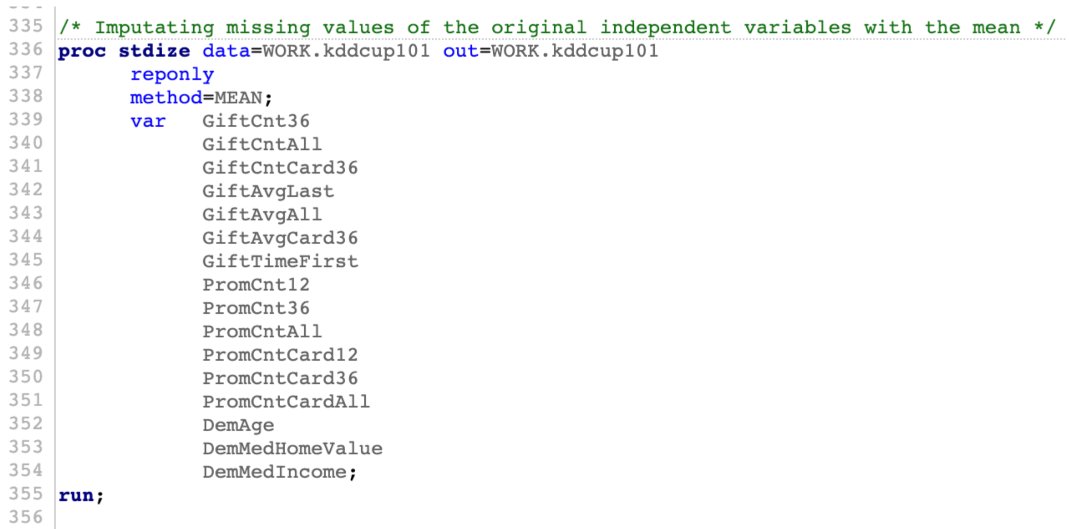


```
/*3.1 missing value imputation for the categorical data*/
data WORK.kddcup100;
    set WORK.kddcup99;
    if StatusCat96NK=' ' then StatusCat96NK_new="missing";
        else StatusCat96NK_new=StatusCat96NK;
    drop StatusCat96NK;
    rename StatusCat96NK_new=StatusCat96NK;
run;

proc freq data=WORK.kddcup100;
    tables  StatusCat96NK;
run;
```

3.2- missing value imputation for the numeric independent variables with missing value indicator

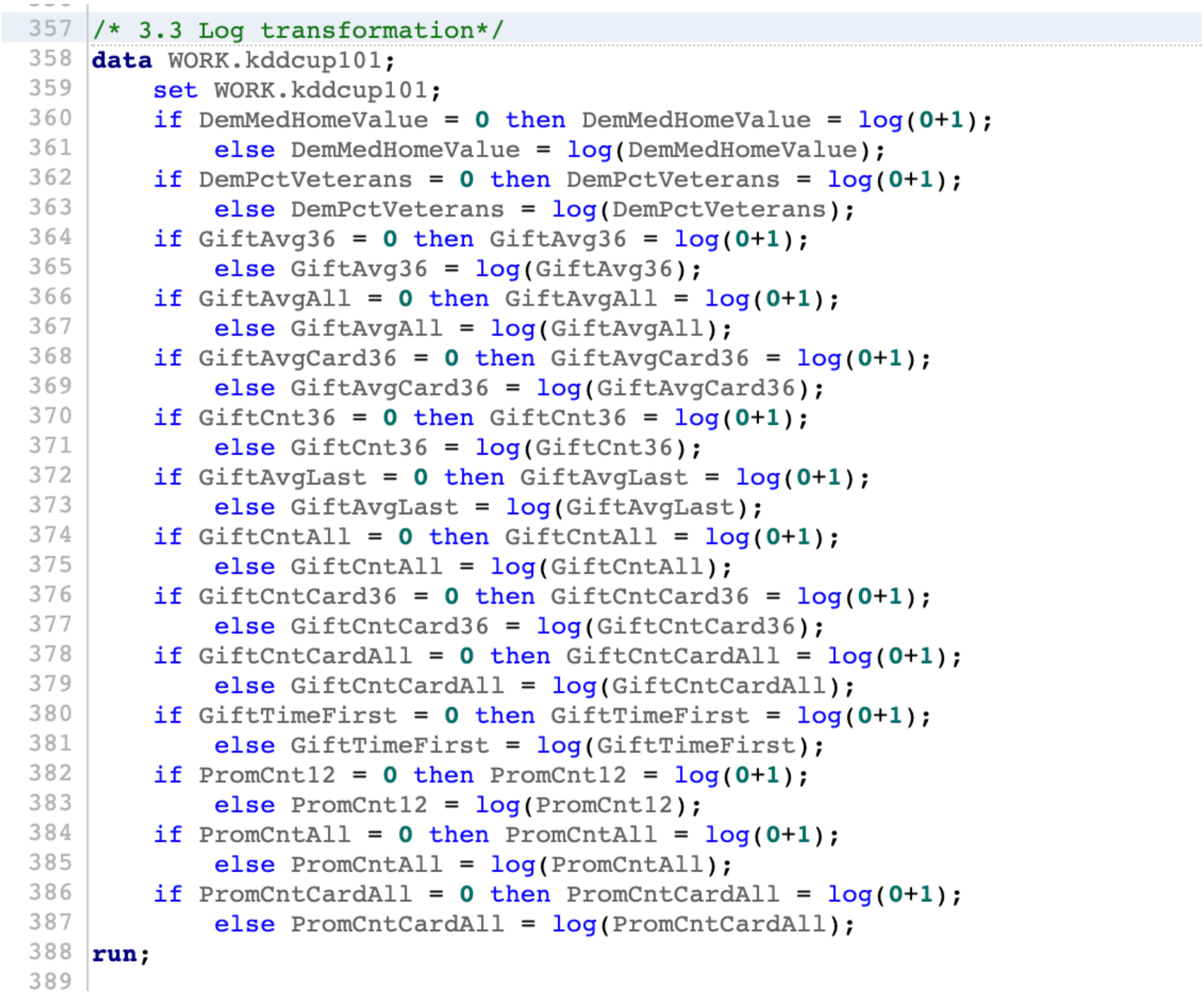


```
/*3.2 missing value imputation for numeric independent variables*/
/*
/*Introducing new missing value indicator variables */
data WORK.kddcup101;
    set WORK.kddcup100;
    if GiftCnt36 =" " then GiftCnt36_missing =1;
        else GiftCnt36_missing = 0;
    if GiftCntAll =" " then GiftCntAll_missing =1;
        else GiftCntAll_missing = 0;
    if GiftCntCard36 =" " then GiftCntCard36_missing =1;
        else GiftCntCard36_missing = 0;
    if GiftAvgLast =" " then GiftAvgLast_missing =1;
        else GiftAvgLast_missing = 0;
    if GiftAvgAll =" " then GiftAvgAll_missing =1;
        else GiftAvgAll_missing = 0;
    if GiftAvgCard36 =" " then GiftAvgCard36_missing =1;
        else GiftAvgCard36_missing = 0;
    if GiftTimeFirst =" " then GiftTimeFirst_missing =1;
        else GiftTimeFirst_missing = 0;
    if PromCnt12 =" " then PromCnt12_missing =1;
        else PromCnt12_missing = 0;
    if PromCnt36 =" " then PromCnt36_missing =1;
        else PromCnt36_missing = 0;
    if PromCntAll =" " then PromCntAll_missing =1;
        else PromCntAll_missing = 0;
    if PromCntCard12 =" " then PromCntCard12_missing =1;
        else PromCntCard12_missing = 0;
    if PromCntCard36 =" " then PromCntCard36_missing =1;
        else PromCntCard36_missing = 0;
    if PromCntCardAll =" " then PromCntCardAll_missing =1;
        else PromCntCardAll_missing = 0;
    if DemAge =" " then DemAge_missing =1;
        else DemAge_missing = 0;
    if DemMedHomeValue =" " then DemMedHomeValue_missing =1;
        else DemMedHomeValue_missing = 0;
    if DemMedIncome =" " then DemMedIncome_missing =1;
        else DemMedIncome_missing = 0;
run;
```



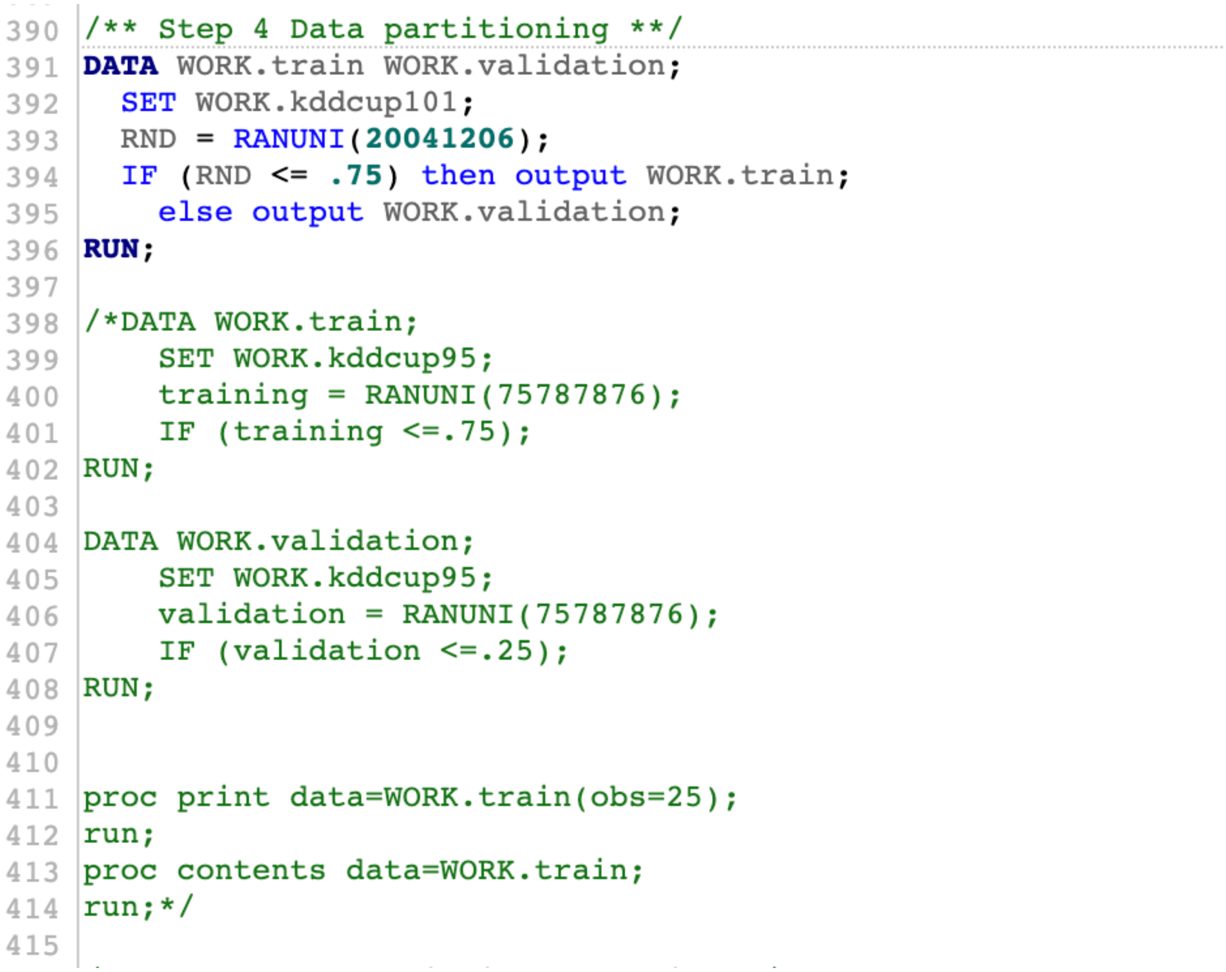
```
/* Imputating missing values of the original independent variables with the mean */
proc stdize data=WORK.kddcup101 out=WORK.kddcup101
      reponly
      method=MEAN;
      var   GiftCnt36
            GiftCntAll
            GiftCntCard36
            GiftAvgLast
            GiftAvgAll
            GiftAvgCard36
            GiftTimeFirst
            PromCnt12
            PromCnt36
            PromCntAll
            PromCntCard12
            PromCntCard36
            PromCntCardAll
            DemAge
            DemMedHomeValue
            DemMedIncome;
run;

```

3.3- Log transformation for the continuous independent variables with right skewed distributions



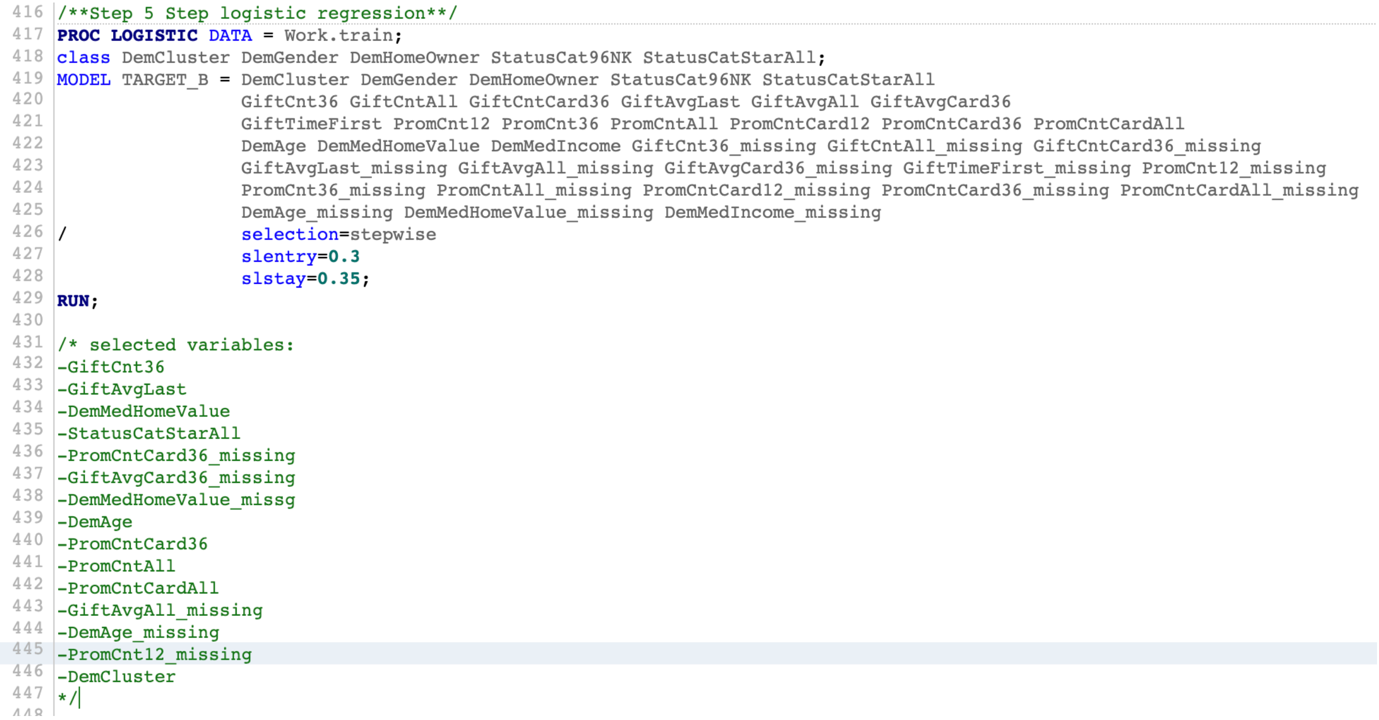
```
/* 3.3 Log transformation*/
data WORK.kddcup101;
    set WORK.kddcup101;
    if DemMedHomeValue = 0 then DemMedHomeValue = log(0+1);
        else DemMedHomeValue = log(DemMedHomeValue);
    if DemPctVeterans = 0 then DemPctVeterans = log(0+1);
        else DemPctVeterans = log(DemPctVeterans);
    if GiftAvg36 = 0 then GiftAvg36 = log(0+1);
        else GiftAvg36 = log(GiftAvg36);
    if GiftAvgAll = 0 then GiftAvgAll = log(0+1);
        else GiftAvgAll = log(GiftAvgAll);
    if GiftAvgCard36 = 0 then GiftAvgCard36 = log(0+1);
        else GiftAvgCard36 = log(GiftAvgCard36);
    if GiftCnt36 = 0 then GiftCnt36 = log(0+1);
        else GiftCnt36 = log(GiftCnt36);
    if GiftAvgLast = 0 then GiftAvgLast = log(0+1);
        else GiftAvgLast = log(GiftAvgLast);
    if GiftCntAll = 0 then GiftCntAll = log(0+1);
        else GiftCntAll = log(GiftCntAll);
    if GiftCntCard36 = 0 then GiftCntCard36 = log(0+1);
        else GiftCntCard36 = log(GiftCntCard36);
    if GiftCntCardAll = 0 then GiftCntCardAll = log(0+1);
        else GiftCntCardAll = log(GiftCntCardAll);
    if GiftTimeFirst = 0 then GiftTimeFirst = log(0+1);
        else GiftTimeFirst = log(GiftTimeFirst);
    if PromCnt12 = 0 then PromCnt12 = log(0+1);
        else PromCnt12 = log(PromCnt12);
    if PromCntAll = 0 then PromCntAll = log(0+1);
        else PromCntAll = log(PromCntAll);
    if PromCntCardAll = 0 then PromCntCardAll = log(0+1);
        else PromCntCardAll = log(PromCntCardAll);
run;

```

Step 4: Data partitioning



```
/** Step 4 Data partitioning **/
DATA WORK.train WORK.validation;
  SET WORK.kddcup101;
  RND = RANUNI(20041206);
  IF (RND <= .75) then output WORK.train;
    else output WORK.validation;
RUN;

/*DATA WORK.train;
    SET WORK.kddcup95;
    training = RANUNI(75787876);
    IF (training <=.75);
RUN;

DATA WORK.validation;
    SET WORK.kddcup95;
    validation = RANUNI(75787876);
    IF (validation <=.25);
RUN;


proc print data=WORK.train(obs=25);
run;
proc contents data=WORK.train;
run;*/
```

Step 5: Stepwise logistic regression for variable selection

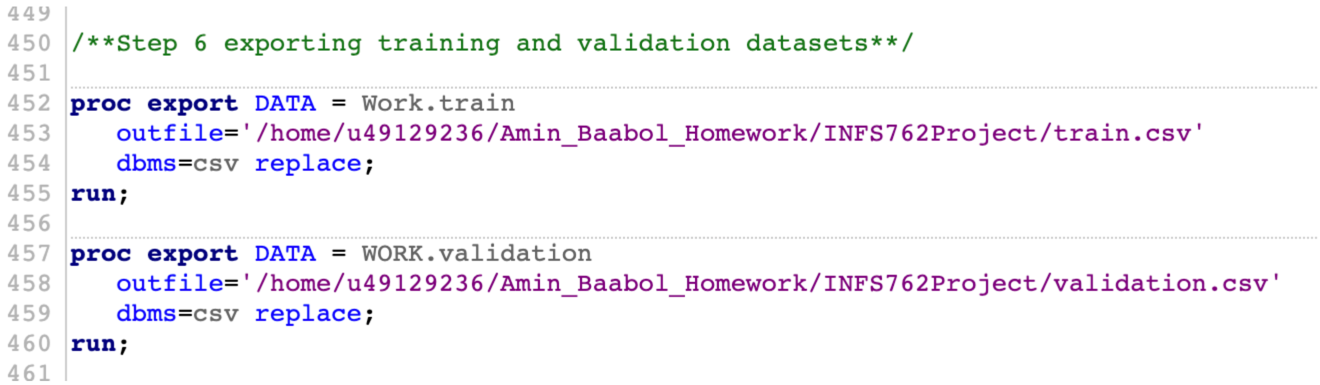


```
/**Step 5 Step logistic regression**/
PROC LOGISTIC DATA = Work.train;
class DemCluster DemGender DemHomeOwner StatusCat96NK StatusCatStarAll;
MODEL TARGET_B = DemCluster DemGender DemHomeOwner StatusCat96NK StatusCatStarAll
                 GiftCnt36 GiftCntAll GiftCntCard36 GiftAvgLast GiftAvgAll GiftAvgCard36
                 GiftTimeFirst PromCnt12 PromCnt36 PromCntAll PromCntCard12 PromCntCard36 PromCntCardAll
                 DemAge DemMedHomeValue DemMedIncome GiftCnt36_missing GiftCntAll_missing GiftCntCard36_missing
                 GiftAvgLast_missing GiftAvgAll_missing GiftAvgCard36_missing GiftTimeFirst_missing PromCnt12_missing
                 PromCnt36_missing PromCntAll_missing PromCntCard12_missing PromCntCard36_missing PromCntCardAll_missing
                 DemAge_missing DemMedHomeValue_missing DemMedIncome_missing
/                selection=stepwise
                 slentry=0.3
                 slstay=0.35;
RUN;

/* selected variables:
-GiftCnt36
-GiftAvgLast
-DemMedHomeValue
-StatusCatStarAll
-PromCntCard36_missing
-GiftAvgCard36_missing
-DemMedHomeValue_missg
-DemAge
-PromCntCard36
-PromCntAll
-PromCntCardAll
-GiftAvgAll_missing
-DemAge_missing
-PromCnt12_missing
-DemCluster
*/
```

Selected explanatory variables list:

- GiftCnt36
- GiftAvgLast
- DemMedHomeValue
- StatusCatStarAll
- PromCntCard36_missing
- GiftAvgCard36_missing
- DemMedHomeValue_missg
- DemAge
- PromCntCard36
- PromCntAll
- PromCntCardAll
- GiftAvgAll_missing
- DemAge_missing
- PromCnt12_missing
- DemCluster

Step 6: Exporting SAS training and validation datasets



```
/**Step 6 exporting training and validation datasets**/

proc export DATA = Work.train
   outfile='/home/u49129236/Amin_Baabol_Homework/INFS762Project/train.csv'
   dbms=csv replace;
run;

proc export DATA = WORK.validation
   outfile='/home/u49129236/Amin_Baabol_Homework/INFS762Project/validation.csv'
   dbms=csv replace;
run;

```

Step 7: Using Weka for logistic regression, neural network and support vector machine; Logistic regression is used for binary classification, the input variables are generally numeric/nominal. This algorithm learns a coefficient for all the explanatory covariates which are then combined into a regression function. Neural network is a classifier that uses backpropagation to learn a multi-layer perceptron to classify instances. Lastly, SVM is also called maximum margin classifier because it draws a line between positive and negative examples and the maximum margin is found which then prevents overfitting.

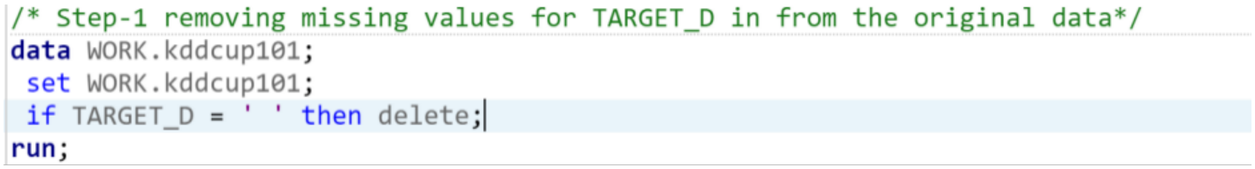
TARGET_B=1	Logistic Regression	Neural Network	SVM(SMO)
Precision	0.567	0.556	0.531
Recall	0.599	0.613	0.578
Accuracy	58.5448%	57.665%	55.0338%

Evidently, all three models have similar accuracies, however, for this particular case logistic regression seems to be outperforming the other two algorithms just slightly. We, therefore, recommend logistic regression.

Task 2: Regression (with TARGET_D as the dependent variable)

Task-2

Step 1 code



```
/* Step-1 removing missing values for TARGET_D in from the original data*/
data WORK.kddcup101;
 set WORK.kddcup101;
 if TARGET_D = ' ' then delete;
run;
```

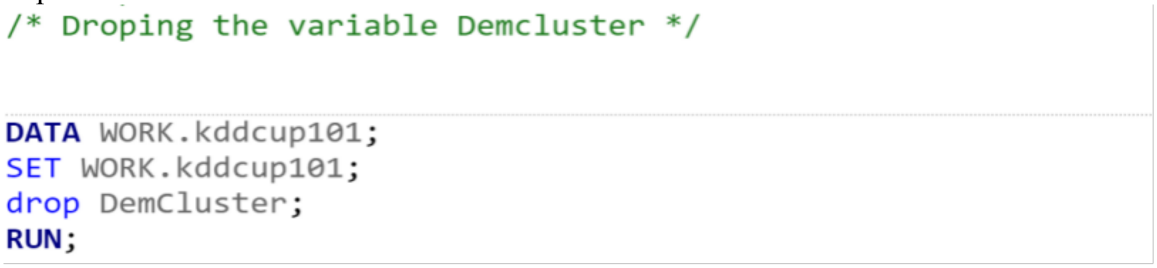
Step 2 (w)

Training dataset is the portion of the data that is used to fit the model. This data is the data that is feed to the model to train and learn.
Validation dataset is the portion of the data used to give an unbiased evaluation of a model and then fine tune parameters. In the industry, validation dataset is used to fine tune the model hyperparameters. The validation dataset result's is used to update parameters.
Test dataset is the portion of the data used to evaluate the model after the model was trained using the train and validation datasets. Usually, this dataset is used to evaluate different models.

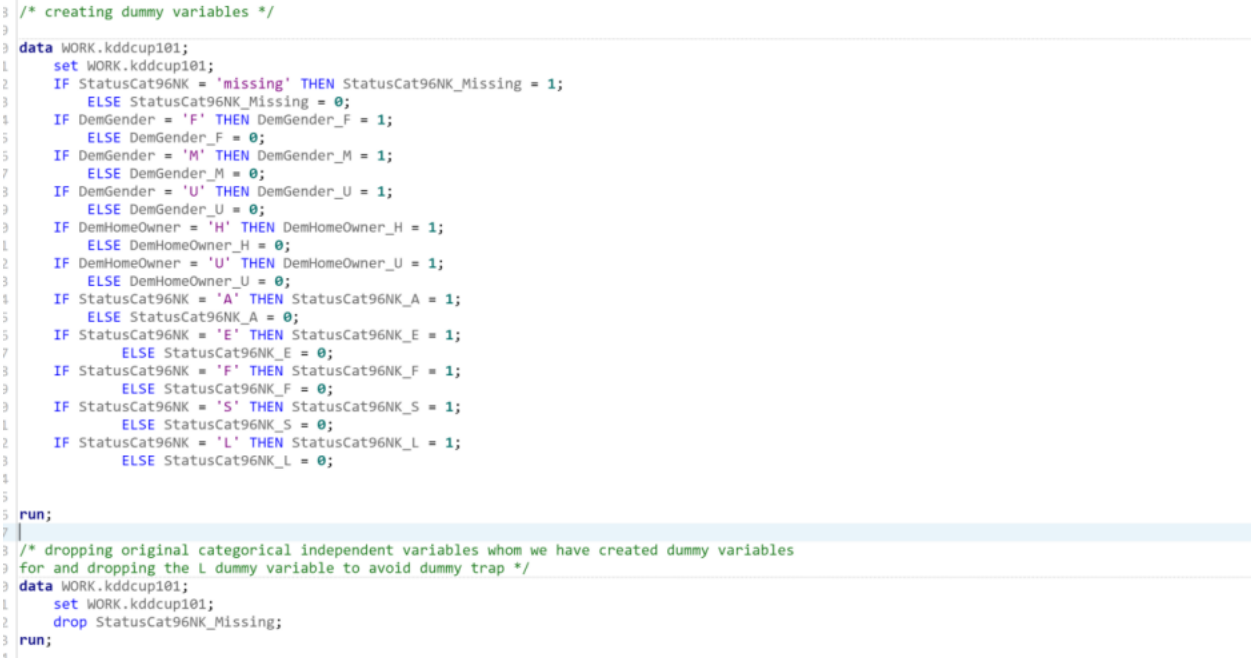
Validation:
For this method, a data set is divided into three data sets training, validation, and testing. We use the training dataset to train the model then we use the validation dataset to test the model and choose the hyperparameters that performed the best on the validation dataset and finally we test the model using the test dataset.

Cross-Validation:
This method divides the dataset into more than one split. It can divide the dataset into 3,5,10 or any k number of splits. The method builds multiple models and for each model it uses some folds train the model and the rest to test the model.

Step 3 code



```
/* Droping the variable Demcluster */


DATA WORK.kddcup101;
SET WORK.kddcup101;
drop DemCluster;
RUN;
```



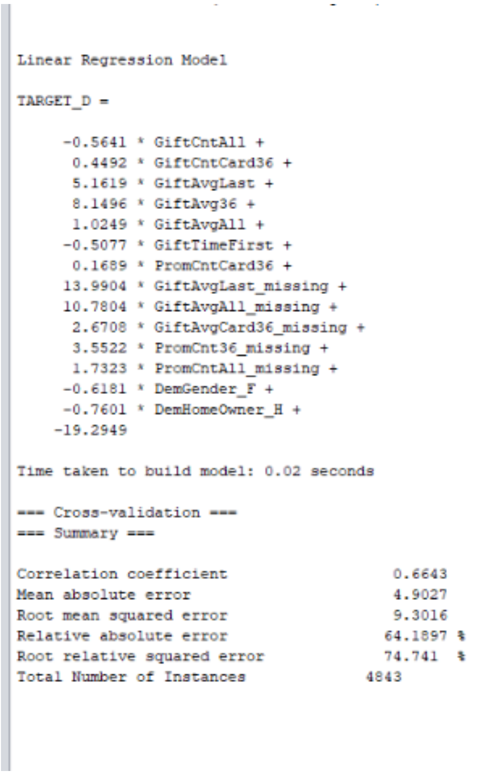
```
/* creating dummy variables */

data WORK.kddcup101;
    set WORK.kddcup101;
    IF StatusCat96NK = 'missing' THEN StatusCat96NK_Missing = 1;
        ELSE StatusCat96NK_Missing = 0;
    IF DemGender = 'F' THEN DemGender_F = 1;
        ELSE DemGender_F = 0;
    IF DemGender = 'M' THEN DemGender_M = 1;
        ELSE DemGender_M = 0;
    IF DemGender = 'U' THEN DemGender_U = 1;
        ELSE DemGender_U = 0;
    IF DemHomeOwner = 'H' THEN DemHomeOwner_H = 1;
        ELSE DemHomeOwner_H = 0;
    IF DemHomeOwner = 'U' THEN DemHomeOwner_U = 1;
        ELSE DemHomeOwner_U = 0;
    IF StatusCat96NK = 'A' THEN StatusCat96NK_A = 1;
        ELSE StatusCat96NK_A = 0;
    IF StatusCat96NK = 'E' THEN StatusCat96NK_E = 1;
            ELSE StatusCat96NK_E = 0;
    IF StatusCat96NK = 'F' THEN StatusCat96NK_F = 1;
            ELSE StatusCat96NK_F = 0;
    IF StatusCat96NK = 'S' THEN StatusCat96NK_S = 1;
            ELSE StatusCat96NK_S = 0;
    IF StatusCat96NK = 'L' THEN StatusCat96NK_L = 1;
            ELSE StatusCat96NK_L = 0;


run;

/* dropping original categorical independent variables whom we have created dummy variables
for and dropping the L dummy variable to avoid dummy trap */
data WORK.kddcup101;
    set WORK.kddcup101;
    drop StatusCat96NK_Missing;
run;
```

Step 4

- GiftAvg36
- GiftAvgLast_missing
- GiftAvgLast
- GiftAvgAll_missing
- PromCnt36_missing
- GiftTimeFirst
- GiftAvgCard36_missing
- PromCntCard36
- DemHomeOwner_H
- DemGender_F
- GiftAvgAll
- PromCntAll_missing
- GiftCntCard36_missing
- PromCntCard12
- DemMedHomeValue_missing
- GiftTimeFirst_missing
- DemPctVeterans
- StatusCat96NK_F
- GiftCntAll
- GiftCntCard36

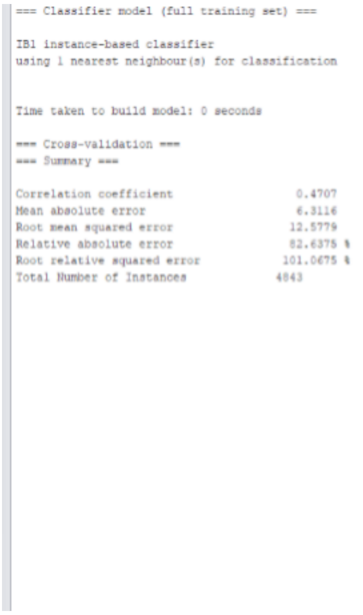
Step 5

1. Linear Regression Model
   RMSE = 9.3016



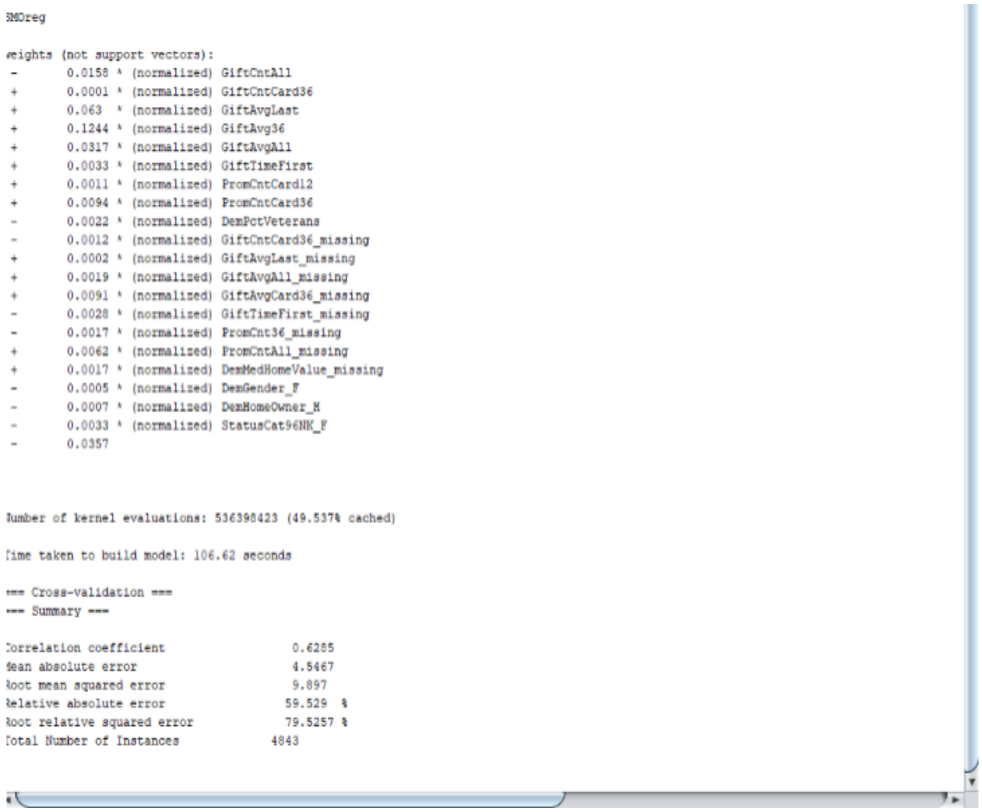
```
Linear Regression Model

TARGET_D =

     -0.5641 * GiftCntAll +
      0.4492 * GiftCntCard36 +
      5.1619 * GiftAvgLast +
      8.1496 * GiftAvg36 +
      1.0249 * GiftAvgAll +
     -0.5077 * GiftTimeFirst +
      0.1689 * PromCntCard36 +
     13.9904 * GiftAvgLast_missing +
     10.7804 * GiftAvgAll_missing +
      2.6708 * GiftAvgCard36_missing +
      3.5522 * PromCnt36_missing +
      1.7323 * PromCntAll_missing +
     -0.6181 * DemGender_F +
     -0.7601 * DemHomeOwner_H +
    -19.2949

Time taken to build model: 0.02 seconds

=== Cross-validation ===
=== Summary ===

Correlation coefficient                  0.6643
Mean absolute error                      4.9027
Root mean squared error                  9.3016
Relative absolute error                 64.1897 %
Root relative squared error             74.741  %
Total Number of Instances             4843
```

2. k nearest neighbor (KNN)
   RMSE = 12.5779



```
=== Classifier model (full training set) ===

IB1 instance-based classifier
using 1 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Cross-validation ===
=== Summary ===

Correlation coefficient                  0.4707
Mean absolute error                      6.3116
Root mean squared error                 12.5779
Relative absolute error                 82.6375 %
Root relative squared error            101.0675 %
Total Number of Instances             4843
```

3. Support Vector Regression
   RMSE = 9.897



```
SMOreg

weights (not support vectors):
 -       0.0158 * (normalized) GiftCntAll
 +       0.0001 * (normalized) GiftCntCard36
 +       0.063  * (normalized) GiftAvgLast
 +       0.1244 * (normalized) GiftAvg36
 +       0.0317 * (normalized) GiftAvgAll
 +       0.0033 * (normalized) GiftTimeFirst
 +       0.0011 * (normalized) PromCntCard12
 +       0.0094 * (normalized) PromCntCard36
 -       0.0022 * (normalized) DemPctVeterans
 -       0.0012 * (normalized) GiftCntCard36_missing
 +       0.0002 * (normalized) GiftAvgLast_missing
 +       0.0019 * (normalized) GiftAvgAll_missing
 +       0.0091 * (normalized) GiftAvgCard36_missing
 -       0.0028 * (normalized) GiftTimeFirst_missing
 -       0.0017 * (normalized) PromCnt36_missing
 +       0.0062 * (normalized) PromCntAll_missing
 +       0.0017 * (normalized) DemMedHomeValue_missing
 -       0.0005 * (normalized) DemGender_F
 -       0.0007 * (normalized) DemHomeOwner_H
 -       0.0033 * (normalized) StatusCat96NK_F
 -       0.0357



Number of kernel evaluations: 536398423 (49.537% cached)

Time taken to build model: 106.62 seconds

=== Cross-validation ===
=== Summary ===

Correlation coefficient                  0.6285
Mean absolute error                      4.5467
Root mean squared error                  9.897
Relative absolute error                 59.529  %
Root relative squared error             79.5257 %
Total Number of Instances             4843
```

which model gives you the best RMSE?)
Linear regression model gives the best RMSE value.