Tarea1 San Martin

April 25, 2023

0.0.1 Integrantes

- Javiera San Martin; jasanmartin2018@udec.cl

x_0

[1]:
```
conda install nbconvert
```

```
Collecting package metadata (current_repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.10.1
  latest version: 23.3.1

Please update conda by running

    $ conda update -n base -c defaults conda



# All requested packages already installed.


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

[42]:
```
!export PATH=/Library/TeX/texbin:$PATH
```

[43]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
import statsmodels.formula.api as smf
import sklearn
import scipy
from scipy.stats import nbinom
import seaborn as sns
```

```
%matplotlib inline
```

[44]:
```
charls2 = pd.read_csv('../data/charls2.csv')
charls2.dropna(inplace=True)
```

[45]:
```
charls2.drinkly = charls2.drinkly.astype(int);
charls2.cesd = charls2.cesd.astype(int);
```

[46]:
```
charls2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 8080 entries, 0 to 9451
Data columns (total 15 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       8080 non-null   int64
 1   cesd      8080 non-null   int64
 2   child     8080 non-null   int64
 3   drinkly   8080 non-null   int64
 4   female    8080 non-null   int64
 5   hrsusu    8080 non-null   float64
 6   hsize     8080 non-null   int64
 7   intmonth  8080 non-null   int64
 8   married   8080 non-null   int64
 9   retage    8080 non-null   int64
 10  retin     8080 non-null   int64
 11  retired   8080 non-null   int64
 12  schadj    8080 non-null   int64
 13  urban     8080 non-null   int64
 14  wealth    8080 non-null   float64
dtypes: float64(2), int64(13)
memory usage: 1010.0 KB
```

Se busca identificar la información de la tabla, viendo si los datos son float o int.

[47]:
```
charls2.describe()
```

[47]:

	age	cesd	child	drinkly	female	\
count	8080.000000	8080.000000	8080.000000	8080.000000	8080.000000	
mean	58.164356	9.112376	2.780074	0.325990	0.535272	
std	9.374956	6.481237	1.397316	0.468773	0.498785	
min	21.000000	0.000000	0.000000	0.000000	0.000000	
25%	51.000000	4.000000	2.000000	0.000000	0.000000	
50%	57.000000	8.000000	3.000000	0.000000	1.000000	
75%	64.000000	13.000000	4.000000	1.000000	1.000000	
max	95.000000	30.000000	10.000000	1.000000	1.000000	

	hrsusu	hsize	intmonth	married	retage	\

```
count  8080.000000  8080.000000  8080.000000  8080.000000  8080.000000
mean      2.566736     3.764233     7.506931     0.879084     1.480817
std       1.788298     1.823838     1.001893     0.326050     4.206412
min       0.000000     1.000000     1.000000     0.000000     0.000000
25%       0.000000     2.000000     7.000000     1.000000     0.000000
50%       3.496508     4.000000     7.000000     1.000000     0.000000
75%       4.025352     5.000000     8.000000     1.000000     0.000000
max       5.123964    16.000000    12.000000     1.000000    37.000000

              retin      retired       schadj        urban          wealth
count  8080.000000  8080.000000  8080.000000  8080.000000     8080.000000
mean      0.163861     0.184035     4.039851     0.212005     1479.488490
std       0.370173     0.387536     3.545666     0.408754    43479.865654
min       0.000000     0.000000     0.000000     0.000000 -1000000.000000
25%       0.000000     0.000000     0.000000     0.000000        0.000000
50%       0.000000     0.000000     4.000000     0.000000      400.000000
75%       0.000000     0.000000     8.000000     0.000000     2200.000000
max       1.000000     1.000000    16.000000     1.000000   900100.000000
```

Se usa describe para poder observar las estadisticas e información de la tabla.

[48]:
```
from IPython.display import display
charls2.reset_index(drop=True, inplace=True);
display(charls2.head(5));
```

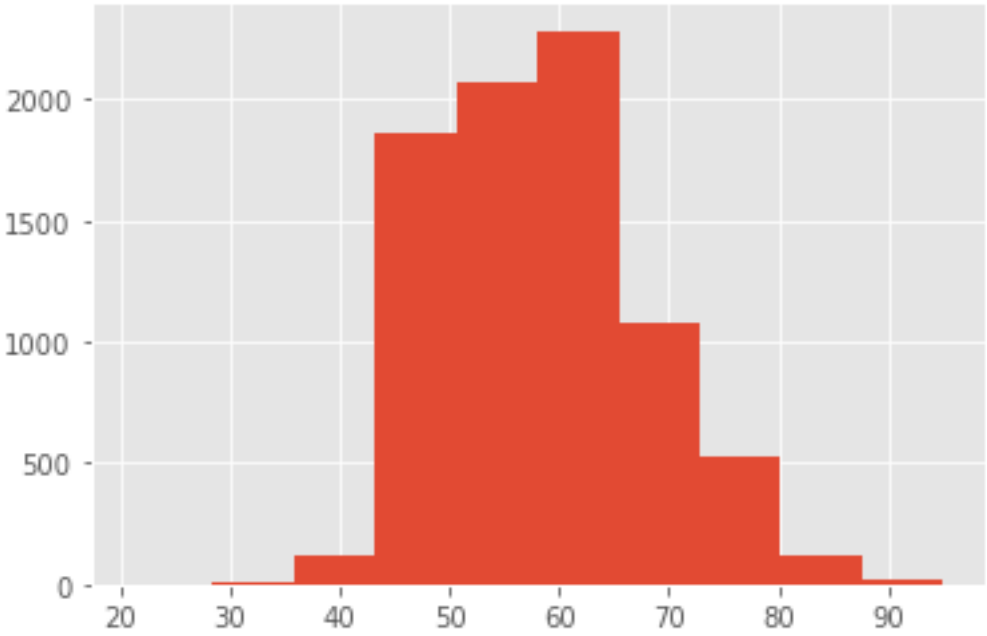
```
   age  cesd  child  drinkly  female    hrsusu  hsize  intmonth  married  \
0   46     6      2        0       1  0.000000      4         7        1
1   48     0      2        1       0  4.143135      4         7        1
2   56     6      1        0       1  0.000000      6         8        1
3   59     6      1        1       0  0.000000      6         8        1
4   47     4      1        1       0  3.806663      3         8        1

   retage  retin  retired  schadj  urban  wealth
0      24      1        0       0      0 -5800.0
1      22      1        0       4      0 -5800.0
2       0      0        0       0      0   350.0
3       0      0        0       0      0   350.0
4      11      1        0       4      0 -8100.0
```

[49]:
```
A="age"
plt.style.use('ggplot')
charls2['age'].hist();
charls2.value_counts(A)
```

[49]:
```
age
48    414
56    373
59    350
```

```
57    348
46    343
      …
22      1
37      1
32      1
30      1
95      1
Length: 61, dtype: int64
```



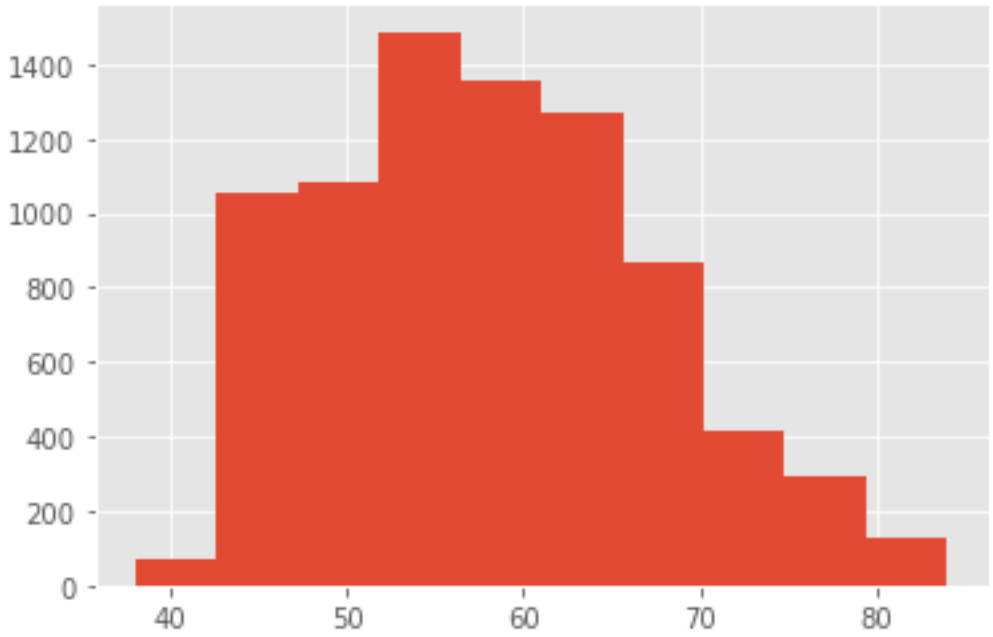
Dentro de los datos de la edad existen datos con muy poca frecuencia por lo que se eliminaran los datos extremos.

[50]:
```
charls2 = charls2[(charls2["age"] < 85) & (charls2["age"] > 37)];
print(len(charls2));
A="age"
plt.style.use('ggplot')
charls2['age'].hist();
charls2.value_counts(A)
```

```
8039
```

[50]:
```
age
48    414
56    373
59    350
```

```
57    348
46    343
58    338
54    329
60    322
47    321
62    301
53    300
55    283
61    283
49    271
45    264
64    260
51    235
63    230
52    203
66    201
65    199
67    188
69    176
50    162
70    152
68    149
71    124
73    121
75     93
72     89
44     83
74     81
76     65
43     47
77     47
78     46
79     44
81     35
80     31
42     27
82     26
83     24
41     16
40     15
84     14
39      8
38      8
dtype: int64
```

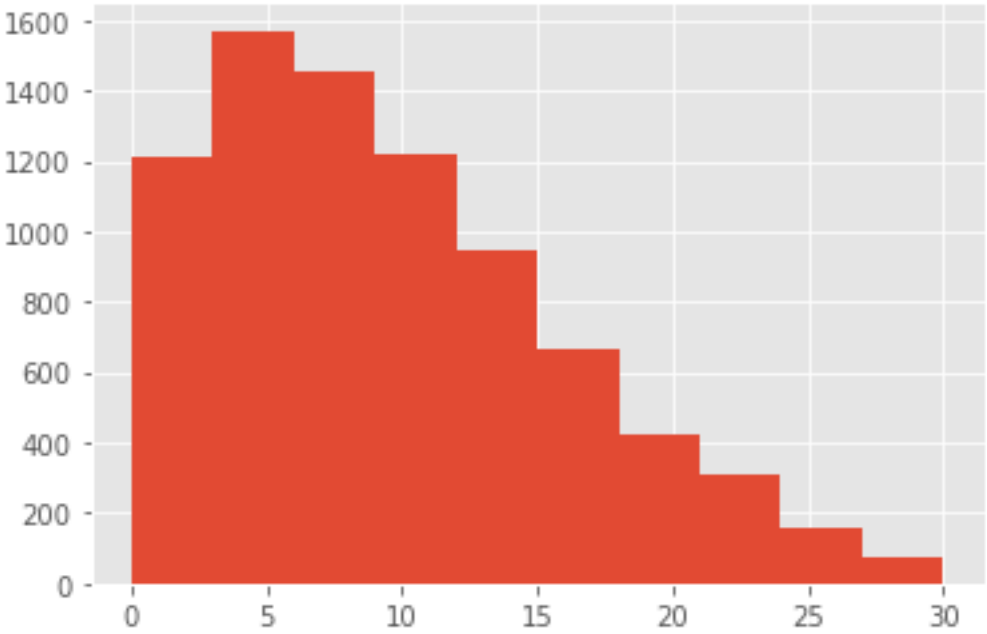


La edad ahora fluctua entre los 37 a 85 años.

[51]:
```
A="cesd"
plt.style.use('ggplot')
charls2['cesd'].hist();
charls2.value_counts(A)
```

[51]:
```
cesd
3     586
6     560
4     500
5     484
0     460
9     456
7     450
8     446
2     403
10    394
12    381
11    370
1     347
14    298
13    269
15    259
16    229
17    178
```

```
18    168
21    137
19    132
20    123
22     87
23     86
24     70
25     52
26     37
27     37
28     14
29     13
30     13
dtype: int64
```

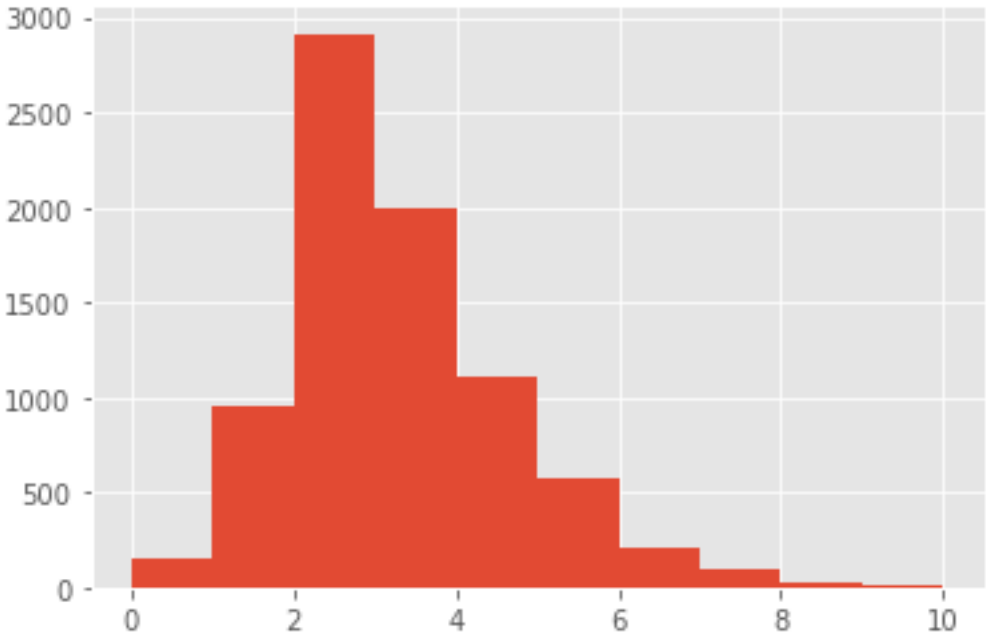


No se eliminarán datos de la variable debido a que ésta representa a los síntomas depresivos en los trabajadores.

[52]:
```
A="child"
plt.style.use('ggplot')
charls2['child'].hist();
charls2.value_counts(A)
```

[52]:
```
child
2     2914
3     1998
```

```
4     1115
1      948
5      579
6      203
0      153
7       90
8       26
9        8
10       5
dtype: int64
```



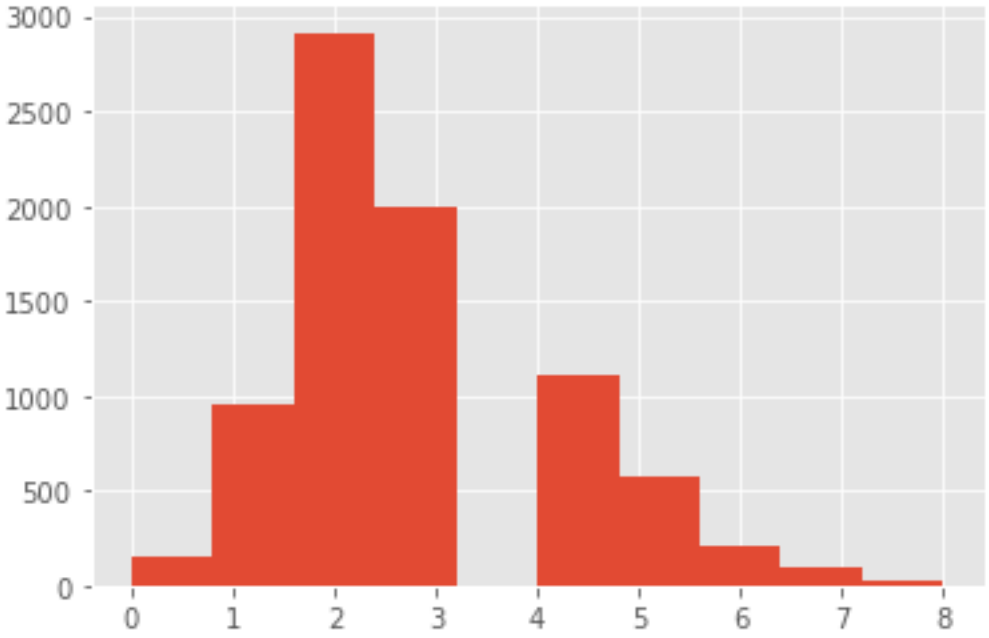
Se eliminarás los datos de “9” y “10” hijos ya que tienen una menor frecuencia.

[53]:
```
charls2 = charls2[(charls2["child"] < 9) & (charls2["child"] >= 0)];
print(len(charls2));
A="child"
plt.style.use('ggplot')
charls2['child'].hist();
charls2.value_counts(A)
```

```
8026
```

[53]:
```
child
2    2914
3    1998
4    1115
```

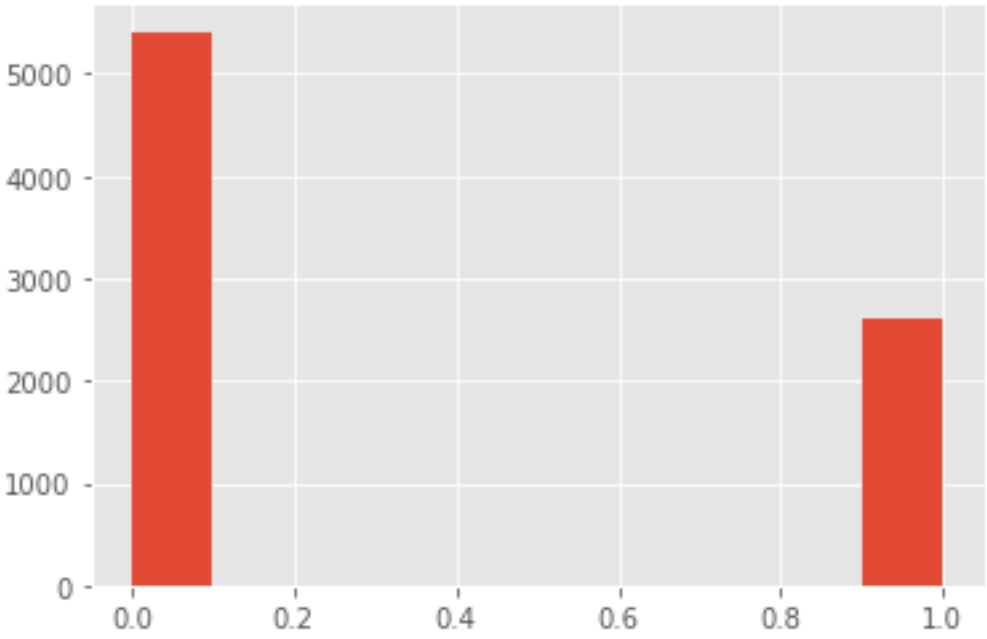
```
1      948
5      579
6      203
0      153
7       90
8       26
dtype: int64
```



Los datos quedan más uniformes.

[54]:
```
A="drinkly"
plt.style.use('ggplot')
charls2['drinkly'].hist();
charls2.value_counts(A)
```

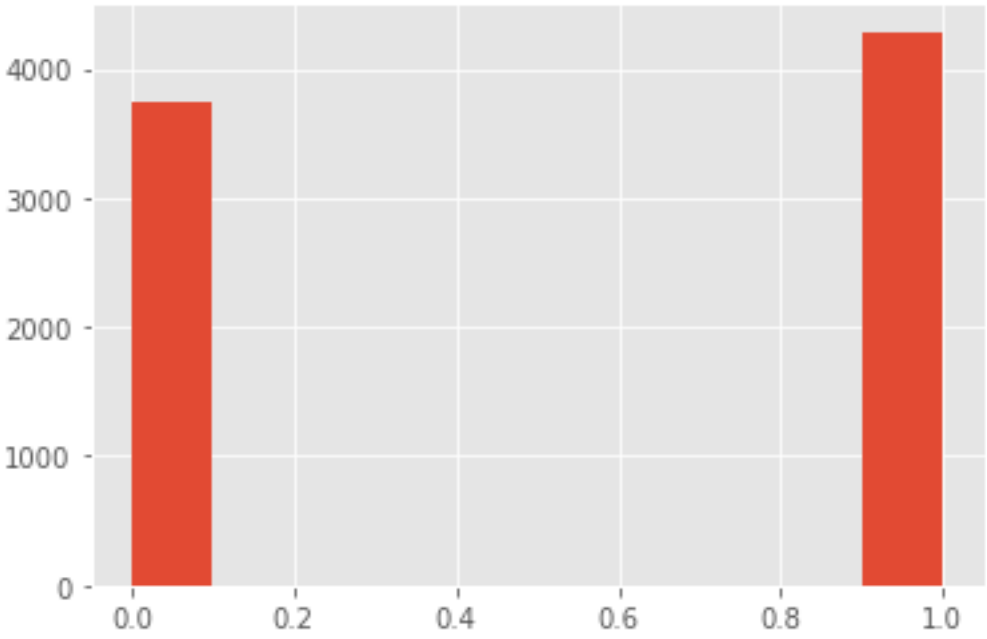
[54]:
```
drinkly
0    5411
1    2615
dtype: int64
```



Al ser una variable binaria no tiene puntos atipicos por lo que se mantiene sin realizarle ninguna modificación.

[55]:
```
A="female"
plt.style.use('ggplot')
charls2['female'].hist();
charls2.value_counts(A)
```

[55]:
```
female
1    4287
0    3739
dtype: int64
```

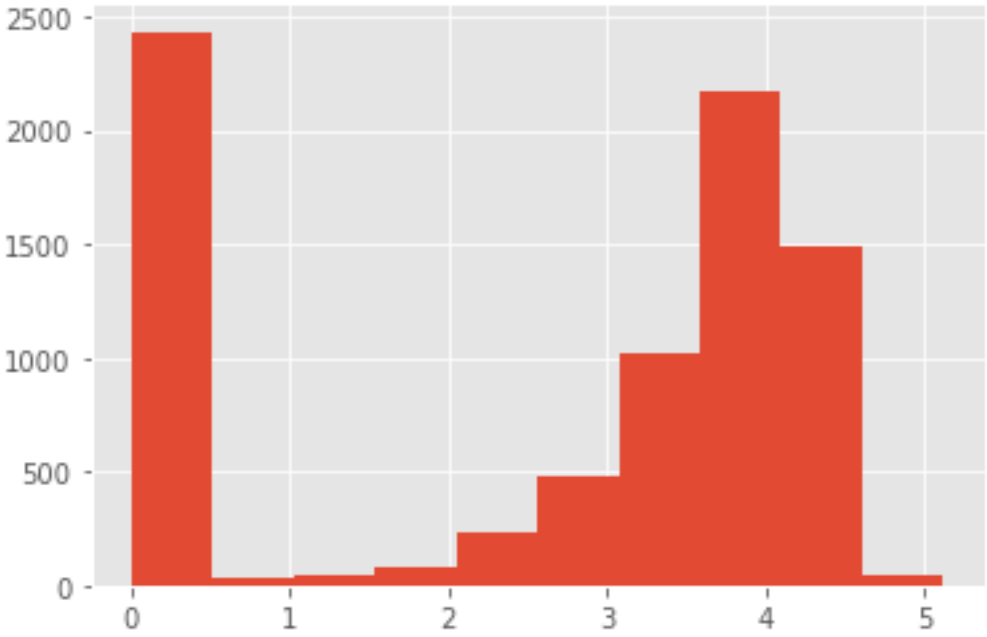


Al ser una variable binaria no tiene puntos atipicos por lo que no se modifica.

[56]:
```
A="hrsusu"
plt.style.use('ggplot')
charls2['hrsusu'].hist();
charls2.value_counts(A)
```

[56]:
```
hrsusu
0.000000    2433
4.025352     925
4.248495     906
3.737670     402
3.555348     251
3.332205     242
4.430817     236
3.688880     220
3.891820     214
4.143135     203
3.871201     164
3.401197     154
3.044522     145
3.178054     128
2.484907     100
4.094345      85
3.912023      83
2.995732      83
```

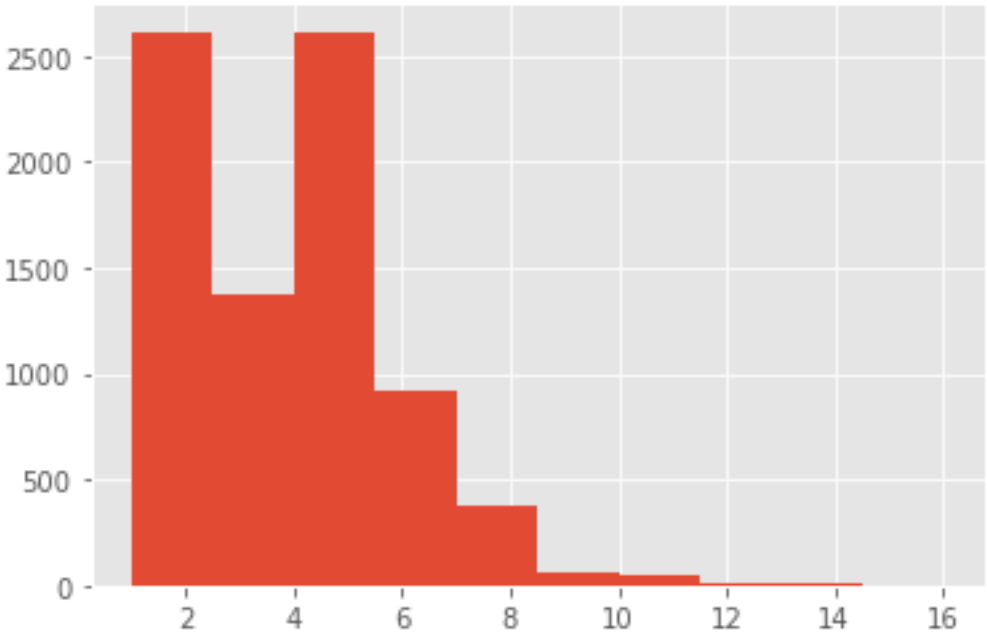
```
3.465736      83
3.218876      77
3.583519      73
4.343805      70
2.772589      69
2.639057      69
2.708050      64
2.890372      52
1.791759      51
2.079442      48
2.302585      46
3.988984      41
2.197225      37
1.386294      32
4.510859      29
0.693147      29
3.806663      21
4.718499      20
4.584968      19
1.945910      19
1.098612      16
4.653960      14
4.276666      13
3.295837      12
4.189655       9
4.007333       8
1.609438       6
3.496508       5
4.356709       4
4.564348       4
3.784190       2
3.663562       2
4.890349       2
4.382027       1
3.951244       1
4.836282       1
4.941642       1
4.969813       1
5.123964       1
dtype: int64
```



Al ser la variable que mide las horas trabajadas en el día no se eliminarán datos.

[57]:
```
A="hsize"
plt.style.use('ggplot')
charls2['hsize'].hist();
charls2.value_counts(A)
```

[57]:
```
hsize
2     2290
3     1380
4     1341
5     1267
6      923
1      325
7      276
8      103
9       64
10      33
11       9
12       7
13       4
14       2
16       2
dtype: int64
```

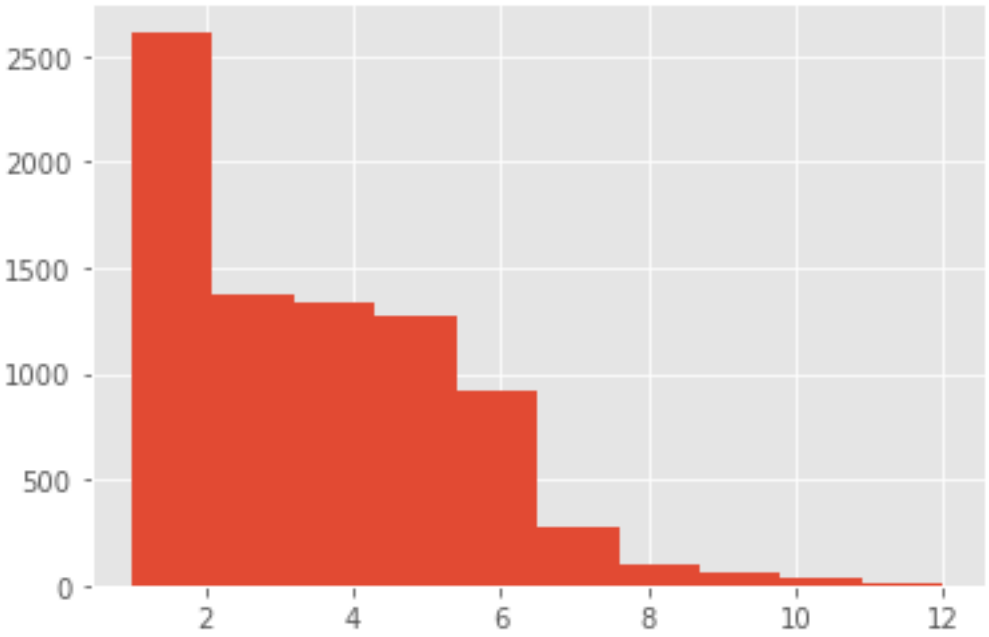


Se ven elementos con poca frecuencia por lo cual se van a eliminar.

[58]:
```
charls2 = charls2[(charls2["hsize"] < 13) & (charls2["hsize"] >= 0)];
print(len(charls2));
A="hsize"
plt.style.use('ggplot')
charls2['hsize'].hist();
charls2.value_counts(A)
```

```
8018
```

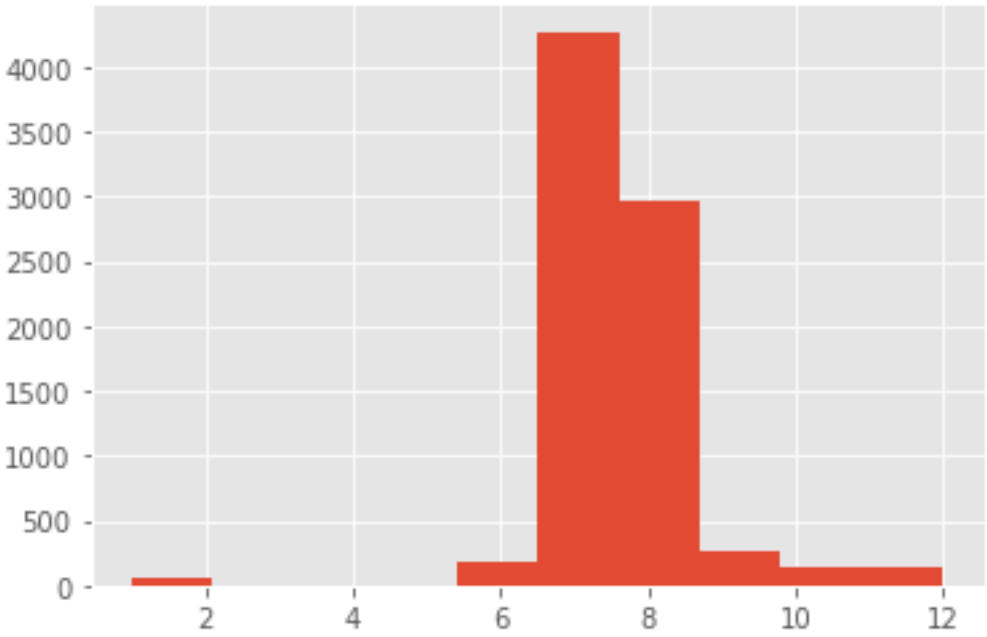
[58]:
```
hsize
2     2290
3     1380
4     1341
5     1267
6      923
1      325
7      276
8      103
9       64
10      33
11       9
12       7
dtype: int64
```



Los datos que han distribuios de una manera más uniforme.

[59]:
```
A="intmonth"
plt.style.use('ggplot')
charls2['intmonth'].hist();
charls2.value_counts(A)
```

[59]:
```
intmonth
7     4269
8     2961
9      269
6      181
10     141
12      95
2       51
11      41
1       10
dtype: int64
```

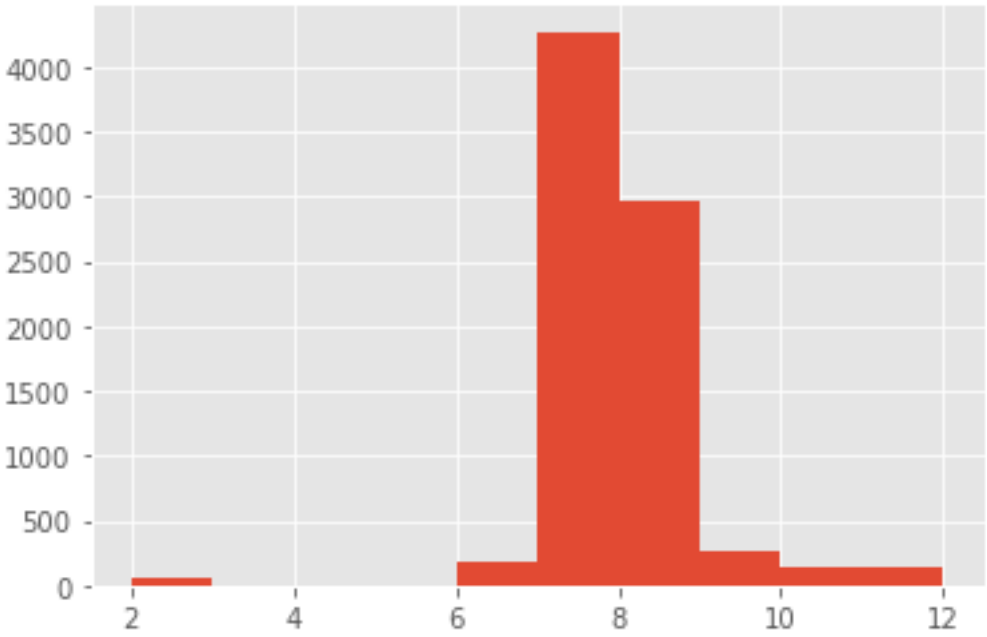


La variable indica el mes en el año que se hizo la encuesta, por lo que se van a eliminar los datos del mes 1

[60]:
```
charls2 = charls2[(charls2["intmonth"] < 13) & (charls2["intmonth"] > 1)];
print(len(charls2));
A="intmonth"
plt.style.use('ggplot')
charls2['intmonth'].hist();
charls2.value_counts(A)
```

```
8008
```

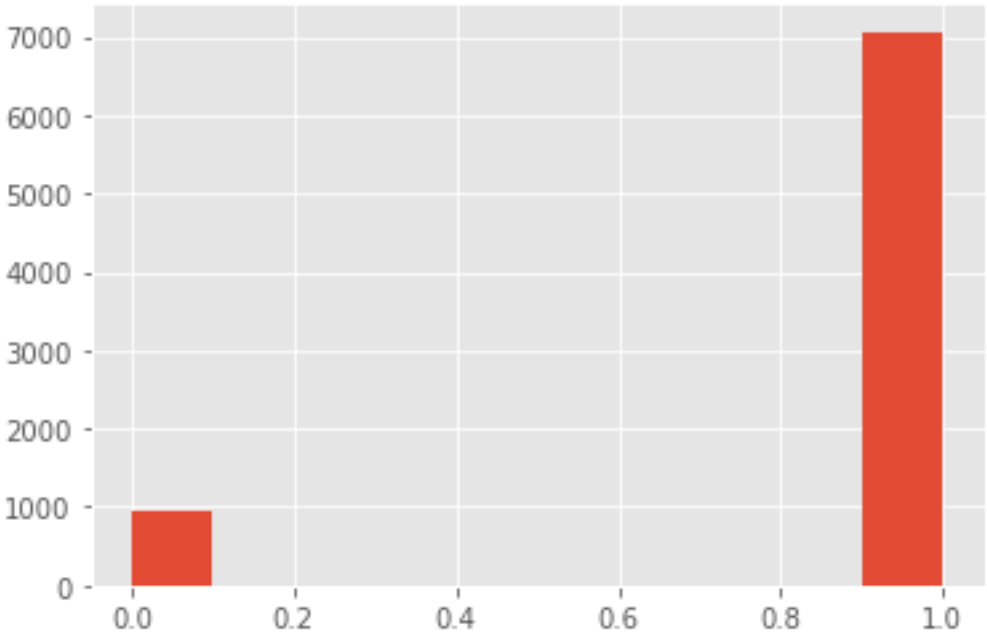
[60]:
```
intmonth
7     4269
8     2961
9      269
6      181
10     141
12      95
2       51
11      41
dtype: int64
```



Se sigue observando una baja frecuencia en el mes 2 y 11 pero al ser una cantidad de datos importantes se van a mantener.

[61]:
```
A="married"
plt.style.use('ggplot')
charls2['married'].hist();
charls2.value_counts(A)
```

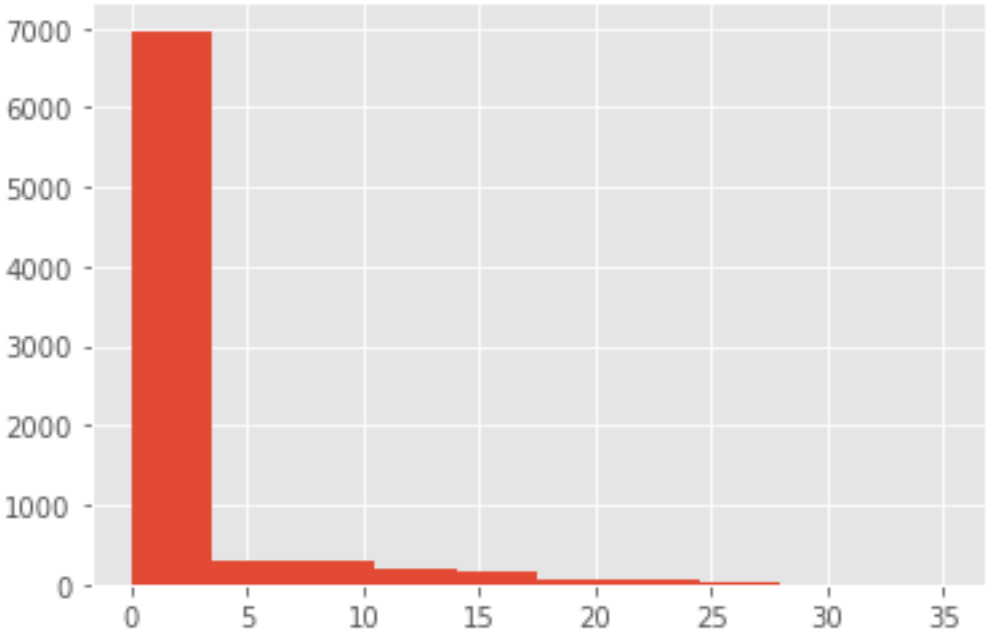
[61]:
```
married
1    7063
0     945
dtype: int64
```



[62]:
```
A="retage"
plt.style.use('ggplot')
charls2['retage'].hist();
charls2.value_counts(A)
```

[62]:
```
retage
0     6687
6      107
2      102
5       95
1       90
3       89
9       85
4       83
8       81
11      68
7       62
12      61
10      56
14      54
13      50
15      47
16      32
17      30
19      20
```

```
22      17
18      17
21      15
20      12
27      10
23       9
24       8
25       6
26       6
30       2
28       1
29       1
31       1
32       1
33       1
34       1
35       1
dtype: int64
```



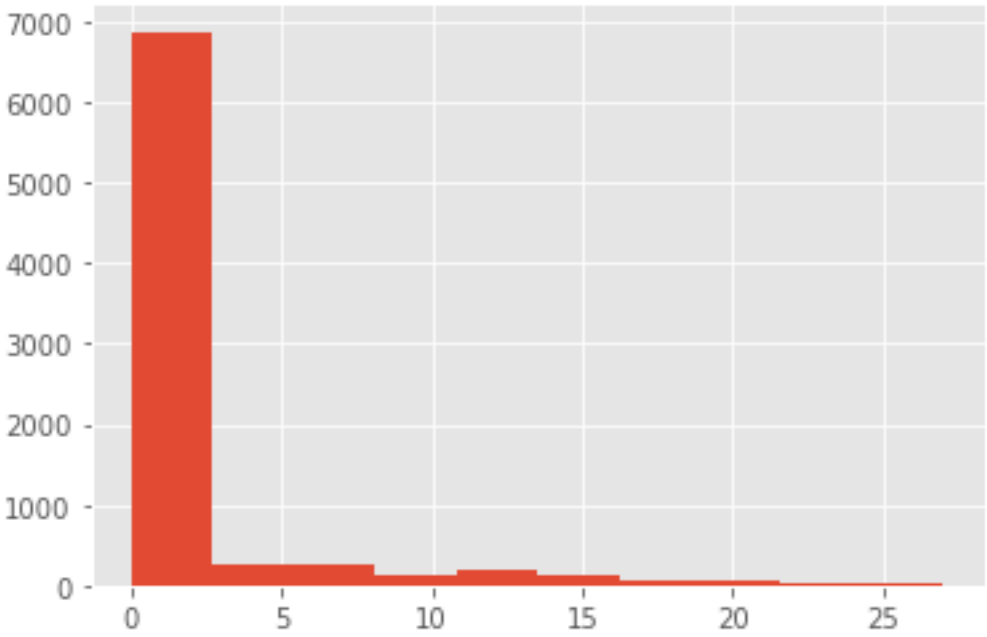
Retage entrega la información de cuando el trabajador se quiere retirar pero al haber varios elementos con baja frecuencia se van a eliminar.

[63]:
```
charls2 = charls2[(charls2["retage"] < 28) & (charls2["retage"] >= 0)];
print(len(charls2));
A="retage"
plt.style.use('ggplot')
```

```
charls2['retage'].hist();
charls2.value_counts(A)
```

7999

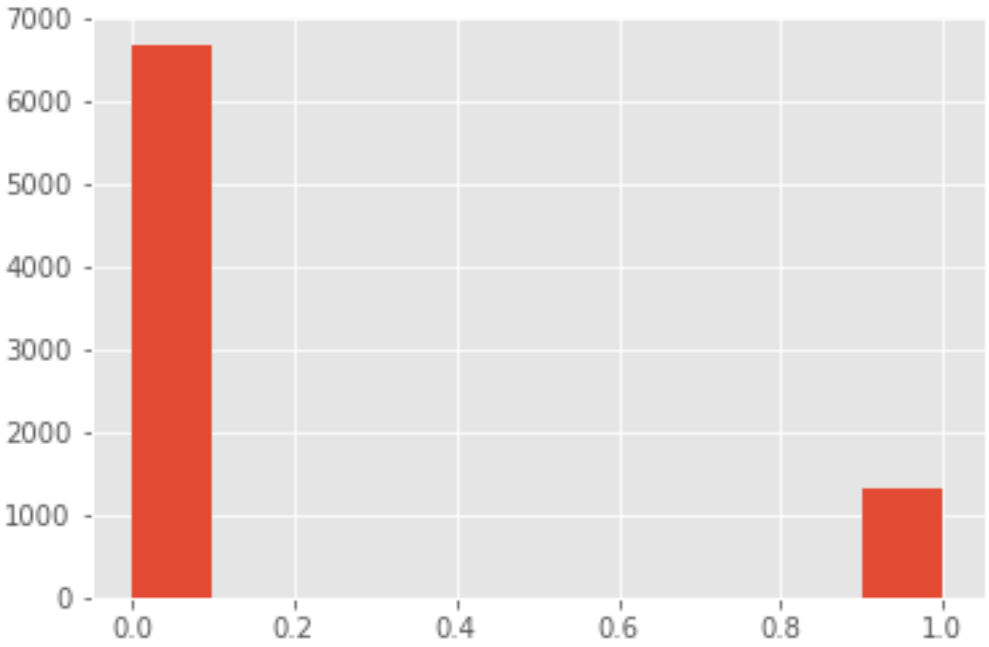
[63]:
```
retage
0     6687
6      107
2      102
5       95
1       90
3       89
9       85
4       83
8       81
11      68
7       62
12      61
10      56
14      54
13      50
15      47
16      32
17      30
19      20
18      17
22      17
21      15
20      12
27      10
23       9
24       8
25       6
26       6
dtype: int64
```



Ahora la variable repreenta mejor los datos.

[64]:
```
A="retin"
plt.style.use('ggplot')
charls2['retin'].hist();
charls2.value_counts(A)
```

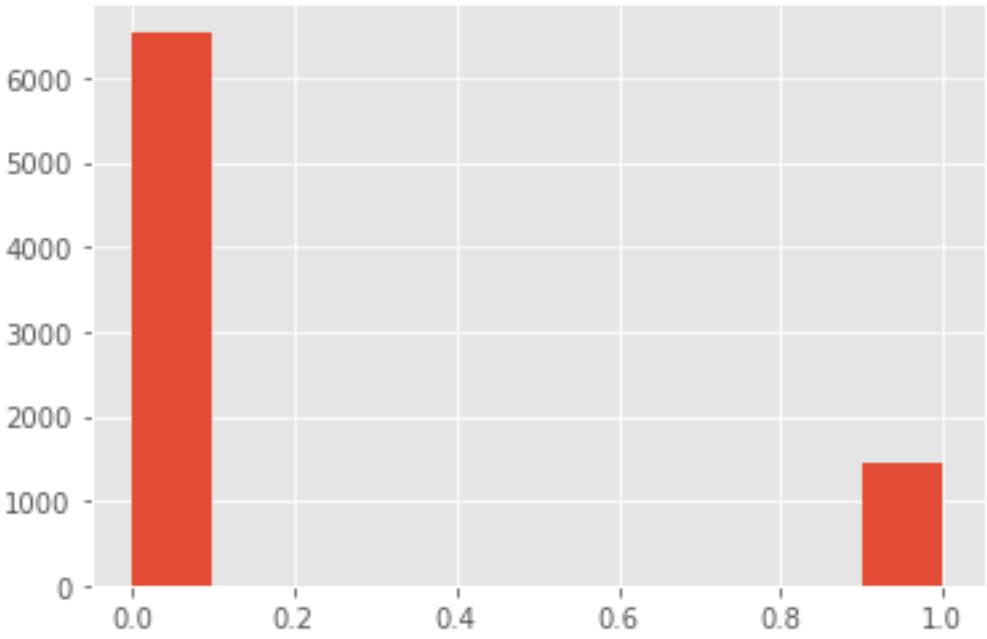
[64]:
```
retin
0    6687
1    1312
dtype: int64
```



La variable representa si esta retirado o no.

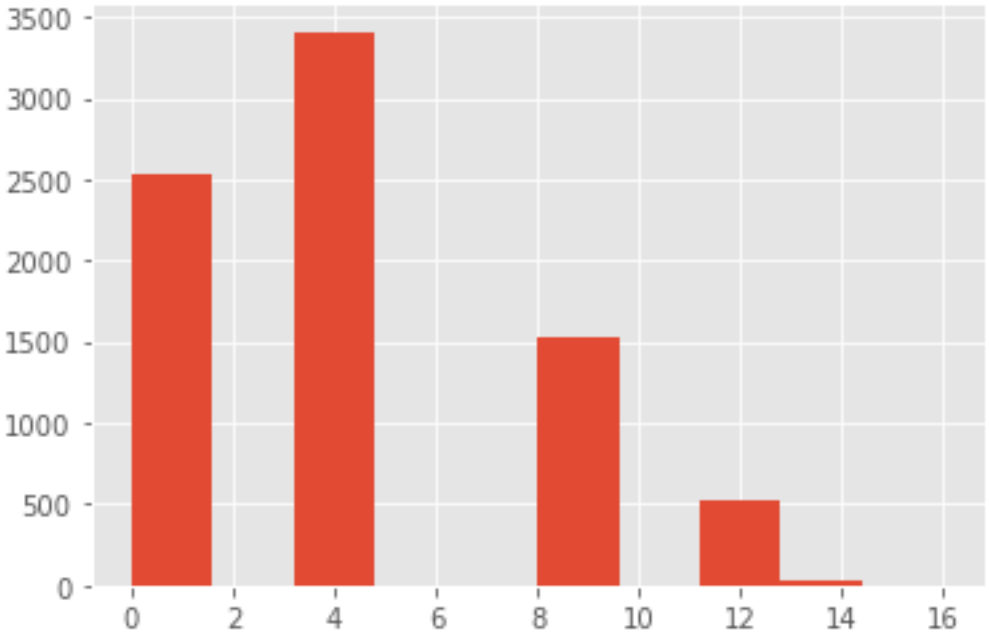
[65]:
```
A="retired"
plt.style.use('ggplot')
charls2['retired'].hist();
charls2.value_counts(A)
```

[65]:
```
retired
0    6544
1    1455
dtype: int64
```



[66]:
```
A="schadj"
plt.style.use('ggplot')
charls2['schadj'].hist();
charls2.value_counts(A)
```

[66]:
```
schadj
4     3408
0     2526
8     1524
12     517
14      23
16       1
dtype: int64
```

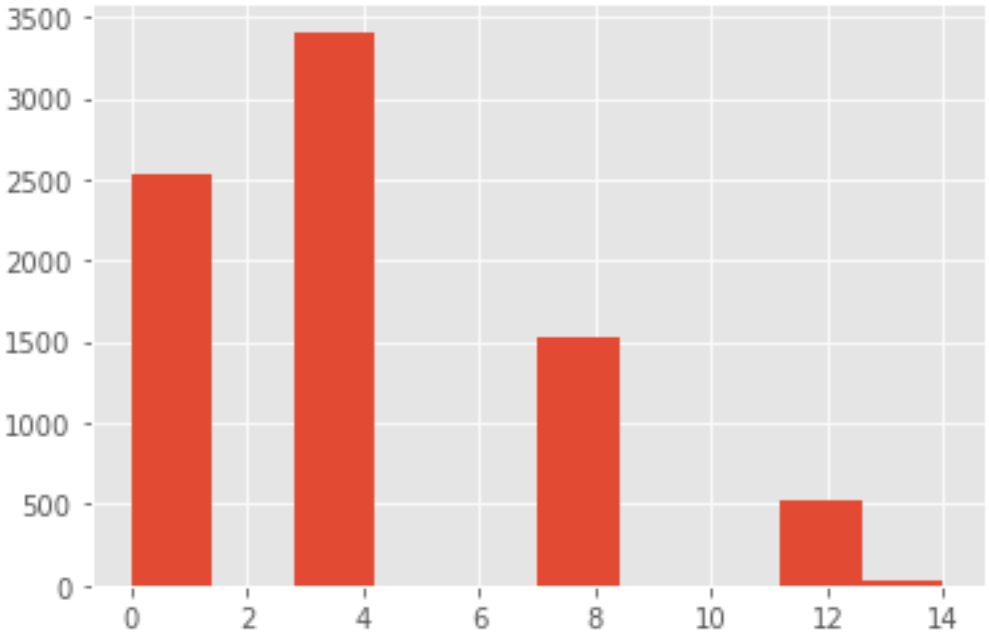


Hay solo una personas que tiene escolaridad hasta los 16 años por lo que se va a eliminar ese dato.

[67]:
```
charls2 = charls2[(charls2["schadj"] < 16) & (charls2["schadj"] >= 0)];
print(len(charls2));
A="schadj"
plt.style.use('ggplot')
charls2['schadj'].hist();
charls2.value_counts(A)
```

```
7998
```

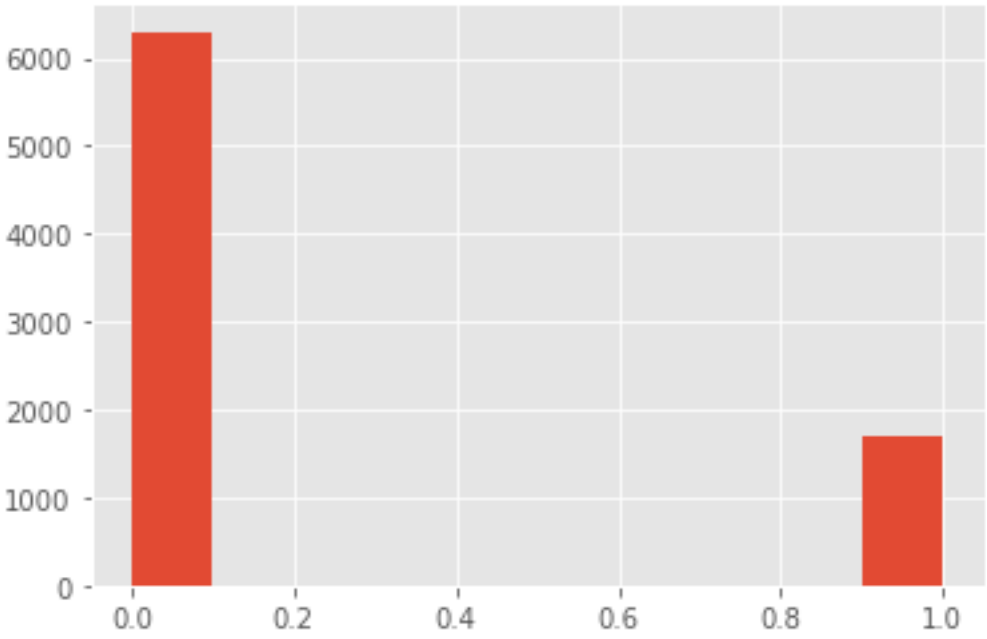
[67]:
```
schadj
4      3408
0      2526
8      1524
12      517
14       23
dtype: int64
```



Se eliminó el dato.

[68]:
```
A="urban"
plt.style.use('ggplot')
charls2['urban'].hist();
charls2.value_counts(A)
```

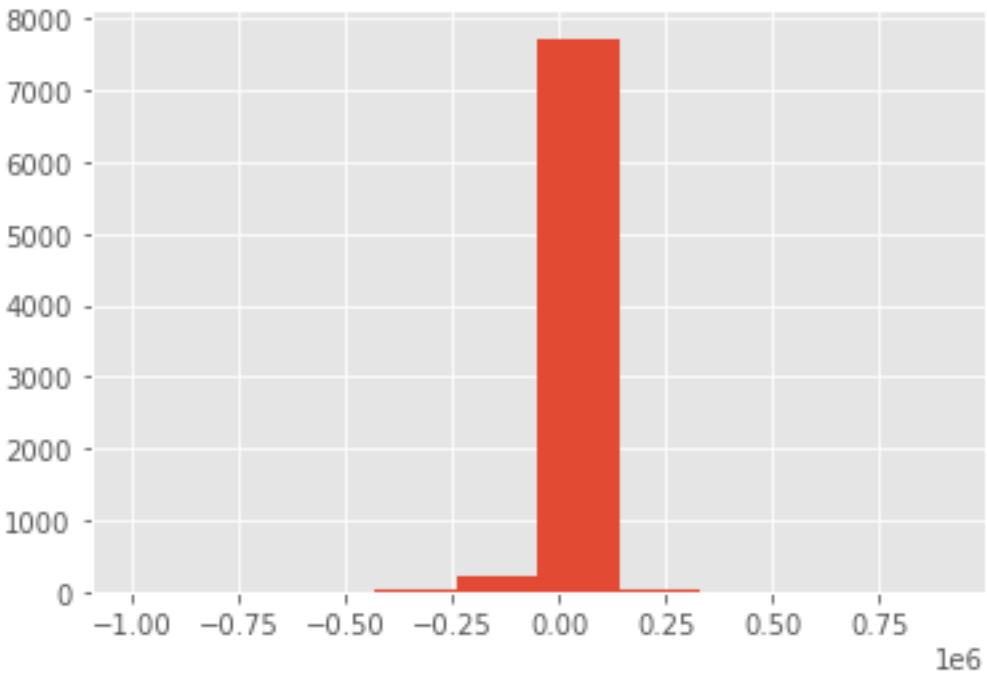
[68]:
```
urban
0    6298
1    1700
dtype: int64
```



Está variables sólo representa si viven en un espacio urbano o no.

[69]:
```
A="wealth"
plt.style.use('ggplot')
charls2['wealth'].hist();
charls2.value_counts(A)
```

[69]:
```
wealth
 0.0          1119
 1000.0        570
 500.0         380
 200.0         357
 100.0         327
               …
-600.0           1
-39350.0         1
-39400.0         1
 31650.0         1
 900100.0        1
Length: 896, dtype: int64
```



La variable riqueza se va a mantener con todos sus datos debido a que puede entregar información importante.

1 2 Mínimos Cuadrados Ordinarios (MCO)

[70]:
```
# Se filtran las filas para tener solo a las personas que no se han retirado.␣
↪Queda 6687 filas.
charls2 = charls2[charls2.loc[:,'retin'] == 0]
charls2.value_counts('retin')
```

[70]:
```
retin
0    6687
dtype: int64
```

[71]:
```
charls2
```

[71]:

	age	cesd	child	drinkly	female	hrsusu	hsize	intmonth	married	\
2	56	6	1	0	1	0.000000	6	8	1	
3	59	6	1	1	0	0.000000	6	8	1	
8	80	17	7	0	1	0.000000	5	7	1	
9	45	4	2	1	0	3.044522	5	7	1	
10	45	0	2	0	1	4.025352	5	7	1	
…	…	…	…	…	…	…	…	…	…	
8073	62	7	4	0	0	2.302585	6	8	1	

```
8075   54     4      3        1        0  0.000000      3         8         1
8076   49     6      3        1        1  3.555348      3         8         1
8077   47    13      2        1        0  0.000000      5         8         1
8079   82     2      6        0        1  4.025352      7         8         0
```

```
      retage  retin  retired  schadj  urban  wealth
2          0      0        0       0      0   350.0
3          0      0        0       0      0   350.0
8          0      0        1       0      0     0.0
9          0      0        0       8      0  5000.0
10         0      0        0       4      0  5000.0
…        …    …        …       …      …     …
8073       0      0        0       8      0  2000.0
8075       0      0        0       8      0   200.0
8076       0      0        0       0      0   200.0
8077       0      0        0       4      0  3850.0
8079       0      0        0       0      0  1000.0

[6687 rows x 15 columns]
```

[75]:
```
y=charls2['retin']
X=charls2[['child','drinkly','married','wealth']];
X=sm.add_constant(X);
model = sm.OLS(y, X);
results = model.fit();
print(results.summary());


# Modelo de Probabilidad Lineal
# 1. interpretar r2
# interpretar f statistic con valor p
# interpretar valor p de coefiecientes
# interpretar valor coef => cambio marginal: -0.0024 => un aumento de un año en␣
 ↪la edad de la persona es una disminucion de la probabilidad de que la␣
 ↪persona se quiera retirar en un 0.2%
# si es binaria: si la persona bebe, aumenta en x%.
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  retin   R-squared:                       nan
Model:                            OLS   Adj. R-squared:                  nan
Method:                 Least Squares   F-statistic:                     nan
Date:                Sun, 16 Apr 2023   Prob (F-statistic):              nan
Time:                        20:47:42   Log-Likelihood:                 inf
No. Observations:                6687   AIC:                           -inf
Df Residuals:                    6685   BIC:                           -inf
Df Model:                           1
Covariance Type:            nonrobust
```

```
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const                0          0        nan        nan           0           0
child                0          0        nan        nan           0           0
==============================================================================
Omnibus:                    16801.399   Durbin-Watson:                     nan
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             2507.625
Skew:                           0.000   Prob(JB):                         0.00
Kurtosis:                       0.000   Cond. No.                         7.62
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

```
/Users/macbookair/opt/anaconda3/lib/python3.8/site-
packages/statsmodels/regression/linear_model.py:1715: RuntimeWarning: invalid
value encountered in double_scalars
  return 1 - self.ssr/self.centered_tss
/Users/macbookair/opt/anaconda3/lib/python3.8/site-
packages/statsmodels/regression/linear_model.py:1804: RuntimeWarning: invalid
value encountered in double_scalars
  return self.mse_model/self.mse_resid
/Users/macbookair/opt/anaconda3/lib/python3.8/site-
packages/statsmodels/regression/linear_model.py:903: RuntimeWarning: divide by
zero encountered in log
  llf = -nobs2*np.log(2*np.pi) - nobs2*np.log(ssr / nobs) - nobs2
/Users/macbookair/opt/anaconda3/lib/python3.8/site-
packages/statsmodels/stats/stattools.py:50: RuntimeWarning: invalid value
encountered in double_scalars
  dw = np.sum(diff_resids**2, axis=axis) / np.sum(resids**2, axis=axis)
```

1.0.1 Probit

[78]:
```
# Probit: Se usa otra funcion, donde se incluye la funcion de distribucion de␣
 ↪probabilidad de X...
# Los cambios marginales se estiman distinto, ver segunda tabla.
model = sm.Probit(y, X)
probit_model = model.fit()
print(probit_model.summary())

mfx = probit_model.get_margeff()
print(mfx.summary())
```

```
---------------------------------------------------------------------------
PerfectSeparationError                    Traceback (most recent call last)
<ipython-input-78-614c66db4852> in <module>
```

```
      2 # Los cambios marginales se estiman distinto, ver segunda tabla.
      3 model = sm.Probit(y, X)
----> 4 probit_model = model.fit()
      5 print(probit_model.summary())
      6

~/opt/anaconda3/lib/python3.8/site-packages/statsmodels/discrete/discrete_model
 ↪py in fit(self, start_params, method, maxiter, full_output, disp, callback,␣
 ↪**kwargs)
   2196     def fit(self, start_params=None, method='newton', maxiter=35,
   2197             full_output=1, disp=1, callback=None, **kwargs):
-> 2198         bnryfit = super().fit(start_params=start_params,

   2199                               method=method,
   2200                               maxiter=maxiter,

~/opt/anaconda3/lib/python3.8/site-packages/statsmodels/discrete/discrete_model
 ↪py in fit(self, start_params, method, maxiter, full_output, disp, callback,␣
 ↪**kwargs)
    225             pass  # TODO: make a function factory to have multiple␣
 ↪call-backs
    226
--> 227         mlefit = super().fit(start_params=start_params,

    228                              method=method,
    229                              maxiter=maxiter,

~/opt/anaconda3/lib/python3.8/site-packages/statsmodels/base/model.py in␣
 ↪fit(self, start_params, method, maxiter, full_output, disp, fargs, callback,␣
 ↪retall, skip_hessian, **kwargs)
    517         warn_convergence = kwargs.pop('warn_convergence', True)
    518         optimizer = Optimizer()
--> 519         xopt, retvals, optim_settings = optimizer._fit(f, score,␣
 ↪start_params,

    520                                                        fargs, kwargs,
    521                                                        hessian=hess,

~/opt/anaconda3/lib/python3.8/site-packages/statsmodels/base/optimizer.py in␣
 ↪_fit(self, objective, gradient, start_params, fargs, kwargs, hessian, method,␣
 ↪maxiter, full_output, disp, callback, retall)
    222
    223         func = fit_funcs[method]
--> 224         xopt, retvals = func(objective, gradient, start_params, fargs,␣
 ↪kwargs,
    225                              disp=disp, maxiter=maxiter,␣
 ↪callback=callback,
    226                              retall=retall, full_output=full_output,
```

```
~/opt/anaconda3/lib/python3.8/site-packages/statsmodels/base/optimizer.py in␣
 ↪_fit_newton(f, score, start_params, fargs, kwargs, disp, maxiter, callback,␣
 ↪retall, full_output, hess, ridge_factor)
    424             history.append(newparams)
    425         if callback is not None:
--> 426             callback(newparams)
    427         iterations += 1
    428     fval = f(newparams, *fargs)  # this is the negative likelihood

~/opt/anaconda3/lib/python3.8/site-packages/statsmodels/discrete/discrete_model
 ↪py in _check_perfect_pred(self, params, *args)
    209                 np.allclose(fittedvalues - endog, 0)):
    210             msg = "Perfect separation detected, results not available"
--> 211             raise PerfectSeparationError(msg)
    212
    213     @Appender(base.LikelihoodModel.fit.__doc__)

PerfectSeparationError: Perfect separation detected, results not available
```

1.0.2 Logit

[12]:
```
# Logit se interpreta igual que probit.
model = sm.Logit(y, X)
logit_model = model.fit()
print(logit_model.summary())

mfx = logit_model.get_margeff()
print(mfx.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.623656
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                   stem   No. Observations:                 9065
Model:                          Logit   Df Residuals:                     9060
Method:                           MLE   Df Model:                            4
Date:                Thu, 21 Jul 2022   Pseudo R-squ.:                0.001816
Time:                        22:34:57   Log-Likelihood:                -5653.4
converged:                       True   LL-Null:                       -5663.7
Covariance Type:            nonrobust   LLR p-value:                 0.0003859
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const         43.8868     23.134      1.897      0.058      -1.455      89.229
puntaje        0.0015      0.001      1.602      0.109      -0.000       0.003
nem           -0.0010      0.001     -1.278      0.201      -0.003       0.001
```

```
vacantes      -0.0021      0.001     -3.712      0.000      -0.003      -0.001
year          -0.0222      0.011     -1.937      0.053      -0.045       0.000
==============================================================================
        Logit Marginal Effects
=====================================
Dep. Variable:                   stem
Method:                          dydx
At:                           overall
==============================================================================
                dy/dx    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
puntaje        0.0003      0.000      1.603      0.109   -7.27e-05       0.001
nem           -0.0002      0.000     -1.279      0.201      -0.001       0.000
vacantes      -0.0005      0.000     -3.721      0.000      -0.001      -0.000
year          -0.0048      0.002     -1.939      0.053      -0.010    5.29e-05
==============================================================================
```

1.0.3 Poisson

[7]:
```
# Notar que aca se mide una variable de conteo: retage => se usan otras o las␣
 ↪mismas variables dependientes en X.
# Se interpreta igual que OLS con los B directo, no como logit ni probit.
poisson=sm.GLM(y,X,family=sm.families.Poisson()).fit()
print(poisson.summary())
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:                ingreso   No. Observations:                 8041
Model:                            GLM   Df Residuals:                     8037
Model Family:                 Poisson   Df Model:                            3
Link Function:                    Log   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:            -2.0395e+05
Date:                Mon, 26 Sep 2022   Deviance:                   3.5249e+05
Time:                        10:16:58   Pearson chi2:                 3.16e+05
No. Iterations:                     6   Pseudo R-squ. (CS):              1.000
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
puntaje        0.0002   3.49e-05      5.285      0.000       0.000       0.000
nem            0.0031   3.05e-05    100.611      0.000       0.003       0.003
vacantes       0.0111   2.13e-05    518.682      0.000       0.011       0.011
year           0.0014      4e-06    348.067      0.000       0.001       0.001
==============================================================================
```

1.0.4 Negative Binomial

[10]:
```
negbin=sm.GLM(y,X,family=sm.families.NegativeBinomial()).fit()
print(negbin.summary())
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:                 ingreso   No. Observations:                 8041
Model:                            GLM   Df Residuals:                     8037
Model Family:        NegativeBinomial   Df Model:                            3
Link Function:                    Log   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -49790.
Date:                Mon, 26 Sep 2022   Deviance:                       2200.2
Time:                        10:49:21   Pearson chi2:                 1.56e+03
No. Iterations:                    12   Pseudo R-squ. (CS):             0.2119
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
puntaje        0.0002      0.000      0.355      0.723      -0.001       0.001
nem            0.0030      0.000      7.527      0.000       0.002       0.004
vacantes       0.0184      0.000     43.479      0.000       0.018       0.019
year           0.0012   5.85e-05     20.824      0.000       0.001       0.001
==============================================================================
```

[10]:
```
print("fitted lambda")
print(negbin.mu)
```

```
fitted lambda
[124.73205222 121.41207007 154.07280363 …  68.35326312  73.99860129
  71.20441307]
```

1.0.5 Test overdispersion

A simple test for overdispersion can be determined with the results of the Poisson model, using the ratio of Pearson chi2 / Df Residuals. A value larger than 1 indicates overdispersion. In the case above (6), data suggets overdispersion.

The Negative Binomial model estimated above is using a value of θ (or $\alpha = 1/\theta$) equal to 1. In order to determine the appropiate value of α, you can estimate a simple regression using the output of the Poisson model:

1. Construct the following variable aux=$[(y - \lambda)^2 - \lambda]/\lambda$
2. Regress the variable aux with λ as the only explanatory variable (no constant)
3. The estimated value is an appropiate guess for $\alpha = 1/\theta$

In the model of the previous section, just use the options on sm.families.NegativeBinomial, in order to manually enter the value of alpha. See example below.

[9]:
```
aux=((y-poisson.mu)**2-poisson.mu)/poisson.mu
auxr=sm.OLS(aux,poisson.mu).fit()
print(auxr.summary())
```

```
                                 OLS Regression Results
================================================================================
=======
Dep. Variable:                 ingreso   R-squared (uncentered):
0.396
Model:                             OLS   Adj. R-squared (uncentered):
0.396
Method:                  Least Squares   F-statistic:
5275.
Date:                 Mon, 26 Sep 2022   Prob (F-statistic):
0.00
Time:                         10:17:03   Log-Likelihood:
-44571.
No. Observations:                8041   AIC:
8.914e+04
Df Residuals:                    8040   BIC:
8.915e+04
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
x1             0.2208      0.003     72.631      0.000       0.215       0.227
==============================================================================
Omnibus:                    10267.981   Durbin-Watson:                   1.652
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          4810468.508
Skew:                           6.647   Prob(JB):                         0.00
Kurtosis:                     122.084   Cond. No.                         1.00
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not
contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

[ ]:
```
si alfa o 1/teta es muy pequeño (no distinto de cero)me quedo con poison y si␣
 ↪es grande me quedo con con la binomial por que hay un grado de sobre␣
 ↪disperción
cual es el metodo más opropiado, porque dejamos las variables que dejamos, se␣
 ↪dejan todas las variables, hay un modelo conceptual, como fundamentar si la␣
 ↪variable esta.
```

prefieron pison o la binomial,

[29]:
```
negbin=sm.GLM(y,X,family=sm.families.NegativeBinomial(alpha=0.22)).fit()
print(negbin.summary())
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:                 ingreso   No. Observations:                 8041
Model:                            GLM   Df Residuals:                     8037
Model Family:        NegativeBinomial   Df Model:                            3
Link Function:                    Log   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -47029.
Date:                Thu, 08 Sep 2022   Deviance:                       9596.2
Time:                        21:53:44   Pearson chi2:                 6.89e+03
No. Iterations:                     13   Pseudo R-squ. (CS):             0.6530
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
puntaje        0.0001      0.000      0.649      0.516      -0.000       0.001
nem            0.0030      0.000     15.906      0.000       0.003       0.003
vacantes       0.0183      0.000     91.448      0.000       0.018       0.019
year           0.0012   2.78e-05     44.007      0.000       0.001       0.001
==============================================================================
```

Tarea 1

Instrucciones

Los resultados de los ejericicios propuestos se deben entregar como un notebook por correo electronico a *juancaros@udec.cl* a mas tardar el dia 11/04/23 hasta las 21:00.

Es importante considerar que el código debe poder ejecutarse en cualquier computadora con la data original del repositorio. Recordar la convencion para el nombre de archivo ademas de incluir en su documento titulos y encabezados por seccion. La data a utilizar es **charls2.csv**.

Las variables tienen la siguiente descripcion:

- retin: 1 si planea planea retirarse
- retage: cuando planea retirarse, medido en años desde la fecha de encuesta (0 implica retirado/a o no planea retirarse)
- cesd: puntaje en la escala de salud mental (0-30)
- child: numero de hijos
- drinkly: bebio alcohol en el ultimo mes (binario)
- hrsusu: horas promedio trabajo diario
- hsize: tamano del hogar
- female: 1 si es mujer, 0 si es hombre
- intmonth: mes en que fue encuestado/a (1-12)
- married: si esta casado/a (binario)
- retired: 1 si esta retirado/a (binario)
- schadj: años de escolaridad

- urban: zona urbana (binario)
- wealth: riqueza neta (miles RMB)
- age: edad al entrar a la encuesta

Preguntas:

1. Cargar la base de datos *charls2.csv* en el ambiente. Identifique los tipos de datos que se encuentran en la base, realice estadisticas descriptivas sobre las variables importantes (Hint: Revisar la distribuciones, datos faltantes, outliers, etc.) y limpie las variables cuando sea necesario.

2. Ejecute un modelo de probabilidad lineal (*MCO*) que permita explicar la probabilidad de que una persona que aun trabaja quiera retirarse (*retin*). Seleccione las variables dependientes a incluir en el modelo final e interprete su significado.

3. Ejecute un modelo *probit* para responder a la pregunta 2. Seleccione las variables dependientes a incluir en el modelo final e interprete su significado.

4. Ejecute un modelo *logit* para responder a la pregunta 2. Seleccione las variables dependientes a incluir en el modelo final e interprete su significado.

5. Comente los resultados obtenidos en 2, 3 y 4. ¿Cuáles y por qué existen las diferencias entre los resultados?. En su opinión, ¿Cuál sería el más adecuado para responder la pregunta de investgación y por qué? ¿Qué variables resultaron ser robustas a la especificación?

6. Ejecute un modelo Poisson para explicar cuando planea retirarse las personas que planean hacerlo. Seleccione las variables dependientes a incluir en el modelo final e interprete su significado.

7. Determine sobre dispersion y posible valor optimo de alpha para un modelo Binomial Negativa.

8. Usando la informacion anterior, ejecute un modelo Binomial Negativa para explicar el número de personas que hay dentro de un hogar. (*n_personas*). Seleccione las variables dependientes a incluir en el modelo final e interprete su significado.

9. Comente los resultados obtenidos en 6, 7 y 8. ¿Cuáles y por qué existen las diferencias entre los resultados?. En su opinión, ¿Cuál sería el más adecuado para responder la pregunta de investgación y por qué? ¿Qué variables resultaron ser robustas a la especificación?

Clase decisión de trasporte, tiene varias opciones que van a tener persistencia entre periodos. Hay que utilizar un estimador robusto de varainza, no significa que (min 14) Hay un probema en que hay correlación y no hay sesgo y hay otra en la que si hay sesgo. t es una variable que tiene cada periodo, teta se mueve de un periodo a otro, (damis ? tentencia?)

Erros se va a descomponer, ese mu i es parte del error, es fijo en el tipo y hay una parte que se mueve en el tiempo. Comuna, tenemos un punto de medición, y hay que ver bien como se mide esa información en un punto que tiene un area deshigual. hay que ver. hay cosas que no deberian cambiar y uno aprovecha eos para poder ajustar. el supuetso basico es que U_i no tienen que estar relacionados con las X_it, este mu puede tener un efecto en comportamiento. que tan mala va a ser la estimación…

El i+d es mucho menos rentable.

si tenemos dos periodos o más se puede diferenciar el modelo, se puede quitare el resaco, como mu es fijo.

el modelo lo que hace… U_I ESTA correlacionada, Y_it (barra) = Y_it - (1/T)*sum de T…

Cov(X_it, U_i) = 0 , U_i distibulle D(o, sigma^2_mu) en vez de solucionar el problema lo eliminamos, la covarianza va a ser muy pequeña y se puede hacer un ajuste. a todas las variables se le quita el promedio, y la parte que varia se le cambia. si no hubiera un componenete observado mu seria cero y la formula seria la perte del efecto fijo que se vio antes, una variable verla antes, la parte de abjaoe. sla varianza total del error. no se ve mu en practica pero el ajute lo que hace es ver la varianza en el tiempo, se estima por minimos cuadrados generalizados, este moldeo queda definodo por la dos ecuacione sla 2 y la 3ra de la diapo, primero se toma un supuesto.

El tearema del sandwich E_ite error total, todos tiene la misma varianza y la I es la matriz de identidad, cuando no es cierto se hace un estimador de la varianza de los betas. esta formula permite ajustar por correlación y heterosetasticidad.