```
In [1]: import pandas as pd
In [2]: df = pd.read_csv('fe_binning.csv')
         df.head()
Out[2]:
               country happiness
          0 Afghanistan
                        3.982855
                Albania
                        4.606651
              Argentina
                        6.697131
                        4.348320
               Armenia
                        7.309061
               Australia
         df.describe()
In [3]:
Out[3]:
                 happiness
          count 143.000000
                  5.404037
          mean
                  1.116106
            std
            min
                  2.701591
           25%
                  4.614304
           50%
                  5.344383
                  6.279204
           75%
           max
                  7.603434
         binned = pd.cut(df['happiness'], bins=[2,4,6,10], labels=['L','M','H'])
         df['happiness_band'] = binned
```

```
In [15]: df.head()
Out[15]:
                 country happiness happiness_band
            0 Afghanistan
                          3.982855
                                                L
                          4.606651
                  Albania
                                                M
                Argentina
                          6.697131
                                                Н
                          4.348320
                                               M
                 Armenia
                                                Н
                 Australia
                          7.309061
In [16]: df['happiness_band'].value_counts()
Out[16]: M
                 80
                 44
                 19
           Name: happiness_band, dtype: int64
           mapping = pd.read_csv('country_region.csv')
In [17]:
In [19]: mapping.head()
Out[19]:
                     country
                                            region
                  Afghanistan
                                         South Asia
            0
                                 Europe & Central Asia
                      Albania
                      Algeria Middle East & North Africa
              American Samoa
                                   East Asia & Pacific
                      Andorra
                                 Europe & Central Asia
           df.head()
In [20]:
Out[20]:
```

```
happiness happiness_band
                  country
            0 Afghanistan
                           3.982855
                                                  L
                   Albania
                           4.606651
                                                 Μ
                 Argentina
                           6.697131
                                                 Н
            3
                           4.348320
                                                 Μ
                  Armenia
                                                 Н
                 Australia
                           7.309061
In [22]: | df = pd.merge(df, mapping, on=['country', 'country'], how='left')
           df.head()
Out[22]:
                  country happiness happiness_band
                                                                    region
                           3.982855
                                                                 South Asia
            O Afghanistan
                                                        Europe & Central Asia
                   Albania
                           4.606651
                                                 M
                 Argentina
                           6.697131
                                                 H Latin America & Caribbean
                           4.348320
                                                        Europe & Central Asia
            3
                  Armenia
                                                 Н
                                                           East Asia & Pacific
                 Australia
                           7.309061
In [23]: df.isnull().mean()
Out[23]: country
                                  0.0
                                  0.0
           happiness
           happiness band
                                  0.0
                                  0.0
           region
           dtype: float64
           df2 = pd.read csv('fe splitting.csv')
In [24]:
           df2.head()
Out[24]:
                           borough
                                                             property_type timestamp_of_call
            0 Kensington And chelsea Purpose Built Flats/Maisonettes - 4 to 9 storeys
                                                                            01/01/2017 16:48
```

borough object property_type object timestamp_of_call object dtype: object 28]: df2['timestamp_of_call'] = pd.to_datetime(df2['timestamp_of_call'] df2.dtypes 28]: borough object property_type object timestamp_of_call datetime64[ns] dtype: object 29]: df2['day'] = df2['timestamp_of_call'].dt.day df2['month'] = df2['timestamp_of_call'].dt.month df2['year'] = df2['timestamp_of_call'].dt.year df2['weekday'] = df2['timestamp_of_call'].dt.weekday df2['hour'] = df2['timestamp_of_call'].dt.hour df2.head()			borough		proper	ty_type	times	tamp_	of_call
3 Westminster Purpose Built Flats/Maisonettes - 4 to 9 storeys 01/01/2017 00:28 4 Barking And dagenham House - single occupancy 01/01/2017 13:33 26]: df2.dtypes 26]: borough object property_type object timestamp_of_call object dtype: object 28]: df2['timestamp_of_call'] = pd.to_datetime(df2['timestamp_of_call df2.dtypes 28]: borough object property_type object timestamp_of_call datetime64[ns] dtype: object 28]: df2['day'] = df2['timestamp_of_call'].dt.day df2['month'] = df2['timestamp_of_call'].dt.month df2['year'] = df2['timestamp_of_call'].dt.year df2['weekday'] = df2['timestamp_of_call'].dt.weekday df2['hour'] = df2['timestamp_of_call'].dt.hour df2.head() 29]: borough property_type timestamp_of_call day month year weekd of the state of the s		1	Camden	Purpose Built Flat	s/Maisonettes - 4 to 9	storeys	01/0	01/2017	7 22:20
4 Barking And dagenham House - single occupancy 01/01/2017 13:33 26]: df2.dtypes 26]: borough object property_type object dtype: object 28]: df2['timestamp_of_call'] = pd.to_datetime(df2['timestamp_of_call df2.dtypes 28]: borough object property_type object timestamp_of_call datetime64[ns] dtype: object 29]: df2['day'] = df2['timestamp_of_call'].dt.day df2['month'] = df2['timestamp_of_call'].dt.weakday df2['year'] = df2['timestamp_of_call'].dt.weekday df2['weekday'] = df2['timestamp_of_call'].dt.weekday df2['hour'] = df2['timestamp_of_call'].dt.hour 29]: borough property_type timestamp_of_call day month year weekd 0 Kensington Purpose Built 2017-01-01 1 2017		2	Southwark	Purpose Built Flat	s/Maisonettes - 4 to 9	storeys	01/0	01/2017	7 09:51
df2.dtypes 26]: df2.dtypes 26]: borough object property_type object timestamp_of_call object dtype: object 28]: df2['timestamp_of_call'] = pd.to_datetime(df2['timestamp_of_call df2.dtypes 28]: borough object property_type object timestamp_of_call datetime64[ns] dtype: object 29]: df2['day'] = df2['timestamp_of_call'].dt.day df2['month'] = df2['timestamp_of_call'].dt.month df2['year'] = df2['timestamp_of_call'].dt.year df2['weekday'] = df2['timestamp_of_call'].dt.weekday df2['hour'] = df2['timestamp_of_call'].dt.weekday df2['hour'] = df2['timestamp_of_call'].dt.hour df2.head() 29]: borough property_type timestamp_of_call day month year weekd of the prop		3	Westminster	Purpose Built Flat	s/Maisonettes - 4 to 9	storeys	01/0	01/2017	7 00:28
borough object property_type object timestamp_of_call object dtype: object 28]: df2['timestamp_of_call'] = pd.to_datetime(df2['timestamp_of_call df2.dtypes 28]: borough object property_type object timestamp_of_call datetime64[ns] dtype: object 29]: df2['day'] = df2['timestamp_of_call'].dt.day df2['month'] = df2['timestamp_of_call'].dt.month df2['year'] = df2['timestamp_of_call'].dt.weekday df2['weekday'] = df2['timestamp_of_call'].dt.weekday df2['hour'] = df2['timestamp_of_call'].dt.hour df2.head() 29]: borough property_type timestamp_of_call day month year weekd Purpose Built O Kensington Flats/Maisonettes - 4 to 9 Purpose Built 16:48:00 1 1 2017		4 Bark	king And dagenham		House - single occ	upancy	01/0	01/2017	7 13:33
property_type object timestamp_of_call object dtype: object df2['timestamp_of_call'] = pd.to_datetime(df2['timestamp_of_call df2.dtypes 28]: df2['dtypes 28]: borough object property_type object timestamp_of_call datetime64[ns] dtype: object 29]: df2['day'] = df2['timestamp_of_call'].dt.day df2['month'] = df2['timestamp_of_call'].dt.month df2['year'] = df2['timestamp_of_call'].dt.year df2['weekday'] = df2['timestamp_of_call'].dt.weekday df2['hour'] = df2['timestamp_of_call'].dt.hour df2.head() 29]: borough property_type timestamp_of_call day month year weekd Purpose Built O Kensington Flats/Maisonettes - 4 to 9 Purpose Built O Kensington Flats/Maisonettes - 4 to 9 Purpose Built O Kensington Flats/Maisonettes - 4 to 9 Purpose Built O Kensington Flats/Maisonettes - 4 to 9	[26]:	df2.dt	ypes						
df2.dtypes 28]: borough	[26]:	proper timest	ty_type amp_of_call	object					
property_type object timestamp_of_call datetime64[ns] dtype: object 29]: df2['day'] = df2['timestamp_of_call'].dt.day df2['month'] = df2['timestamp_of_call'].dt.month df2['year'] = df2['timestamp_of_call'].dt.year df2['weekday'] = df2['timestamp_of_call'].dt.weekday df2['hour'] = df2['timestamp_of_call'].dt.hour df2.head() 29]: borough property_type timestamp_of_call day month year weekd Purpose Built 2017-01-01 1 2017 16:48:00 1 2017 16:48:00 1 2017 16:48:00 1 2017 16:48:00 1 2017 16:48:00 1 2017 16:48:00 1 2017 16:48:00 1	[28]:	_		call'] = pd.t	o_datetime(df2	['tim	nestam	np_of	_call
<pre>df2['month'] = df2['timestamp_of_call'].dt.month df2['year'] = df2['timestamp_of_call'].dt.year df2['weekday'] = df2['timestamp_of_call'].dt.weekday df2['hour'] = df2['timestamp_of_call'].dt.hour df2.head()</pre> <pre> borough</pre>	t[28]:	proper timest	ty_type amp_of_call	ob	ject				
borough property_type timestamp_of_call day month year weekd Purpose Built And chelsea Flats/Maisonettes - 4 to 9 Purpose Built 16:48:00 1 1 2017	[29]:	df2['m df2['y df2['w df2['h	<pre>onth'] = df2 ear'] = df2[eekday'] = d our'] = df2[</pre>	['timestamp_o 'timestamp_of f2['timestamp	f_call'].dt.mo _call'].dt.yea _of_call'].dt.	r weekd	lay		
0 Kensington Flats/Maisonettes - 4 to 9 2017-01-01 1 1 2017	[29]:		borough	property_type	timestamp_of_call	day	month	year	weekd
				Maisonettes - 4 to 9		1	1	2017	

		borough	property_type	timestamp_of_call	day	month	year	weekday	hour
	1	Camden	Purpose Built Flats/Maisonettes - 4 to 9 storeys	2017-01-01 22:20:00	1	1	2017	6	22
	2	Southwark	Purpose Built Flats/Maisonettes - 4 to 9 storeys	2017-01-01 09:51:00	1	1	2017	6	9
	3	Westminster	Purpose Built Flats/Maisonettes - 4 to 9 storeys	2017-01-01 00:28:00	1	1	2017	6	0
	4	Barking And dagenham	House - single occupancy	2017-01-01 13:33:00	1	1	2017	6	13
In [34]:	df2	.isnull().	mean()						
Out[34]:	proptime day monyea week hou	r kday	0.0 0.0 0.0 0.0 0.0						
In [36]:	df2	.head()							
Out[36]:		borough	property_type	timestamp_of_call	day	month	year	weekday	hour
	0	Kensington And chelsea	Purpose Built Flats/Maisonettes - 4 to 9 storeys	2017-01-01 16:48:00	1	1	2017	6	16
	1	Camden	Purpose Built Flats/Maisonettes - 4 to 9 storeys	2017-01-01 22:20:00	1	1	2017	6	22

		borough	property_type	timestamp_of_call	day	month	year	weekday	hour			
	2	Southwark	Purpose Built Flats/Maisonettes - 4 to 9 storeys	2017-01-01 09:51:00	1	1	2017	6	9			
	3	Westminster	Purpose Built Flats/Maisonettes - 4 to 9 storeys	2017-01-01 00:28:00	1	1	2017	6	0			
	4	Barking And dagenham	House - single occupancy	2017-01-01 13:33:00	1	1	2017	6	13			
n [37]:	df2['property_type'].unique()											
	lice	'House 'Conve 'Purpo 'Purpo 'Conve 'Self 'Unlice 'House 'Stude 'Unlice 'Nursi 'House 'Hotel 'Hoste 'Licen 'Bunga	se Built Flats/Mai - single occupanc rted Flat/Maisonet se Built Flats/Mai se Built Flats/Mai rted Flat/Maisonet contained Sheltere ensed House in Mul in Multiple Occup nt Hall of Residen ensed House in Mul ng/Care Home/Hospi in Multiple Occup /motel', "Children l (e.g. for homele sed House in Multi low - single occup ary/barracks', 'Ho sed House in Multi ing House/B&B for an/Mobile home (pe	y', te - Up to 2 s sonettes - Up sonettes - 10 tes - 3 or mor d Housing', tiple Occupati ation - 3 or m ce', 'Other Re tiple Occupati ce', "Nurses'/ ation - Up to 's Home", ss people)', ' ple Occupation ancy', 'Other useboat (perma ple Occupation homeless/asylu	tore to 3 or m e st on - ore side on - Doct 2 st Reti - U Dwel nent - 3 m se	eys', store storeys Up to storeys ential 3 or storeys rement Up to 2 coreys define to 2 coreys	eys', torey , 2 s /s (n Home more accom (not t/Old 2 sto , ling) ore s ',	toreys', ot known ', storeys modation known i Persons reys', ', toreys',	if ', '', f li Hom			

```
].str.split('-', expand=True)
           df2.head()
Out[38]:
                             property_type timestamp_of_call day month year weekday hour property_1
                  borough
                              Purpose Built
                                                                                                   Pur
                                                 2017-01-01
                Kensington
                           Flats/Maisonettes
                                                                     1 2017
                                                                                    6
                                                                                        16
               And chelsea
                                                   16:48:00
                                                                                               Flats/Ma
                             - 4 to 9 storeys
                              Purpose Built
                                                 2017-01-01
                                                                                                  Pur
            1
                  Camden Flats/Maisonettes
                                                              1
                                                                     1 2017
                                                                                    6
                                                                                         22
                                                   22:20:00
                                                                                               Flats/Ma
                             - 4 to 9 storeys
                              Purpose Built
                                                 2017-01-01
                                                                                                  Pur
                Southwark Flats/Maisonettes
                                                              1
                                                                     1 2017
                                                                                    6
                                                                                          9
                                                   09:51:00
                                                                                               Flats/Ma
                             - 4 to 9 storeys
                              Purpose Built
                                                 2017-01-01
                                                                                                  Pur
            3 Westminster Flats/Maisonettes
                                                              1
                                                                     1 2017
                                                                                    6
                                                   00:28:00
                                                                                               Flats/Ma
                             - 4 to 9 storeys
                             House - single
               Barking And
                                                 2017-01-01
                                                                     1 2017
                                                                                    6
                                                                                        13
                                                   13:33:00
                 dagenham
                                occupancy
In [39]:
           df2.isnull().mean()
Out[39]: borough
                                      0.00000
                                      0.00000
           property type
           timestamp of call
                                      0.00000
                                      0.00000
           day
                                      0.00000
           month
                                      0.00000
           year
           weekday
                                      0.00000
                                      0.00000
           hour
                                      0.00000
           property type type
           property type size
                                      0.10007
           dtype: float64
In [40]:
           df3 = pd.read csv('fe one hot.csv')
           df3.head()
```

```
Out[40]:
                    country happiness
                                                           region
             0 Afghanistan
                              3.982855
                                                       South Asia
                              4.606651
                                             Europe & Central Asia
                     Albania
                              6.697131 Latin America & Caribbean
                  Argentina
                              4.348320
                                             Europe & Central Asia
                    Armenia
                                                East Asia & Pacific
                   Australia
                              7.309061
```

In [41]: region_one_hot = pd.get_dummies(df3.region)
 region_one_hot.head()

Out[41]:

	East Asia & Pacific	Europe & Central Asia	Latin America & Caribbean	Middle East & North Africa	North America	South Asia	Sub- Saharan Africa
0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0
2	0	0	1	0	0	0	0
3	0	1	0	0	0	0	0
4	1	0	0	0	0	0	0

In [42]: df3 = df3.join(region_one_hot).drop('region', axis=1)
 df3.head()

Out[42]:

	country	happiness	East Asia & Pacific	Europe & Central Asia	Latin America & Caribbean	Middle East & North Africa	North America	South Asia	Sub- Saharan Africa
0	Afghanistan	3.982855	0	0	0	0	0	1	0
1	Albania	4.606651	0	1	0	0	0	0	0
2	Argentina	6.697131	0	0	1	0	0	0	0

```
Middle
                                            Europe
                                     East
                                                        Latin
                                                                                          Sub-
                                                                 East &
                                                                          North South
                                                                                       Saharan
                 country happiness
                                   Asia &
                                                    America &
                                            Central
                                                                  North
                                                                       America
                                                                                 Asia
                                   Pacific
                                                    Caribbean
                                                                                        Africa
                                                                 Africa
                                              Asia
           3
                 Armenia
                          4.348320
                                        0
                                                1
                                                           0
                                                                     0
                                                                             0
                                                                                    0
                                                                                            0
                          7.309061
                                        1
                                                0
                                                           0
                                                                     0
                                                                             0
                                                                                    0
                                                                                            0
                Australia
In [44]:
          df4 = pd.read csv('fe calculated.csv')
          df4.head()
Out[44]:
                            gdp_usd population
                 country
           0 Afghanistan 1.936297e+10
                                      37172386
                 Albania 1.505888e+10
                                       2866376
                Argentina 5.184750e+11
                                      44494502
           3
                 Armenia 1.243309e+10
                                       2951776
                Australia 1.432200e+12
                                      24992369
In [47]:
          df4['gdp usd'] / df4['population']
Out[47]: 0
                     520.896603
                    5253.630064
          2
                   11652.563276
                    4212.070943
                   57305.491928
          125
                   17277.970110
          126
                    1532.371639
          127
                    2563.816070
          128
                    1539.900158
          129
                    2146.996385
          Length: 130, dtype: float64
          df4['per capita'] = df4['gdp usd'] / df4['population']
In [48]:
          df4.head()
```

```
Out[48]:
                                        population
                  country
                              gdp_usd
                                                      per_capita
            0 Afghanistan 1.936297e+10
                                         37172386
                                                     520.896603
                  Albania 1.505888e+10
                                          2866376
                                                    5253.630064
                 Argentina 5.184750e+11
                                         44494502
                                                  11652.563276
            3
                  Armenia 1.243309e+10
                                          2951776
                                                    4212.070943
                 Australia 1.432200e+12
                                         24992369 57305.491928
 In [ ]:
```