ENIF24_debt

August 3, 2025

1 ENIF 2024: Informal debt

1.1 Table of contents

1.2 Introduction

In this work we study the results from the National Survet on Financial Inclusion (ENIF), conducted by the National Banking Comission (CNBV) and the Statistics and Geography National Institute (INEGI), in 2024, which tries to "diagnose, design public policies, and establish goald regarding inclusion and financial education; similarly, it tries to suggest changes and updates to attend to new requirements and considerations on the National Policy of Financial Inclusion".

For this analysis, we focus on Section 4, Subsection 8, of said document, on financial attitudes, behaviour, vulnerability, and over-all well-being and, more especifically, on informal debt.

[1]:
```
### Basic packages

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

### ML models

from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
```

```python
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier

### ML tools

from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.metrics import mean_absolute_error, mean_squared_error,␣
↪root_mean_squared_error
```

1.3 Data loading

The tables are available at:

[ ]:
```python
URL = 'https://www.inegi.org.mx/contenidos/programas/enif/2024/microdatos/
↪enif_2024_bd_csv.zip'
```

The relevant data can be found on the following document.

[2]:
```python
data = pd.read_csv('./TMODULO.csv')
data.head(5)
```

[2]:
```
   LLAVEMOD  LLAVEVIV  LLAVEHOG  EDAD_V  NIV  GRA  P3_1A  P3_2  P3_3  P3_4  \
0    101101       101      1011      38    8    4      2     5     2     2
1    210101       210      2101      36    8    5      2     2     2     2
2    303102       303      3031      20    3    3      2     6     2     2
3    401101       401      4011      59    3    3      2     4     2     2
4    503101       503      5031      37    6    3      2     5     2     2

   …  FOLIO  VIV_SEL  HOGAR  N_REN  SEXO  TLOC  REGION  EST_DIS  UPM_DIS  \
0  …      1        1      1      1     1     1       1       17      196
1  …      2       10      1      1     1     3       6      179     2088
2  …      3        3      1      2     1     1       4       50      763
3  …      4        1      1      1     1     1       1       17      182
4  …      5        3      1      1     2     1       4       49      631

   FAC_PER
0     1233
1     1763
2      903
3      720
4     8114

[5 rows x 398 columns]
```

[3]:
```python
### Subsection
```

```
numss = 8

### The relevant data for Subsectiob 4.6 are extracted

data1 = data[['LLAVEMOD', 'SEXO', 'EDAD_V', 'NIV', 'GRA', 'REGION', 'TLOC',␣
↪f'P4_{numss}_1',f'P4_{numss}_2',f'P4_{numss}_3',f'P4_{numss}_4',f'P4_{numss}_5',f'P4_{numss
data1.head()
```

[3]:

```
   LLAVEMOD  SEXO  EDAD_V  NIV  GRA  REGION  TLOC  P4_8_1  P4_8_2  P4_8_3  \
0    101101     1      38    8    4       1     1       1       1       1
1    210101     1      36    8    5       6     3       3       2       1
2    303102     1      20    3    3       4     1       2       3       3
3    401101     1      59    3    3       1     1       1       1       2
4    503101     2      37    6    3       4     1       3       3       2

   P4_8_4  P4_8_5  P4_8_6
0       1       1       2
1       1       1       2
2       2       1       1
3       2       1       1
4       1       3       3
```

We can see that most of the data are discreet, esentially categorical. We consider sex, age, education, ans region/locality as independent variables, and the answers to all of the questions as the dependent variables. Later, we will define a continuous metric as an aggregate of some of these variables, as to facilitate analysis and interpretation.

Back to the introduction

1.4 Cleaning and integrity

We check for duplicates and NAs, and we verify that all of the inputs are within the ranges specified by INEGI.

[4]:

```
### data types

data1.dtypes
```

[4]:

```
LLAVEMOD    int64
SEXO        int64
EDAD_V      int64
NIV         int64
GRA         int64
REGION      int64
TLOC        int64
P4_8_1      int64
P4_8_2      int64
P4_8_3      int64
```

```
P4_8_4      int64
P4_8_5      int64
P4_8_6      int64
dtype: object
```

We are dealing with numeric data only.

Now we look for duplicates and NAs.

[5]:
```
print(f"Raw data: {data1.shape[1]} columns and {data1.shape[0]} rows")
print(f"Withous NAs: {data1.dropna().shape[1]} columns and {data1.dropna().
 ↪shape[0]} rows")
print(f"Without duplicates: {data1.drop_duplicates().shape[1]} columns and␣
 ↪{data1.drop_duplicates().shape[0]} rows")
```

```
Raw data: 13 columns and 13502 rows
Withous NAs: 13 columns and 13502 rows
Without duplicates: 13 columns and 13502 rows
```

We conclude that the data was pretty clean to begin with. Now we check the ranges for each column.

Sex: Male (1), Female (2).

[6]:
```
pd.DataFrame(data1['SEXO'].value_counts()).sort_values('SEXO').head()
```

[6]:
```
      count
SEXO
1      6082
2      7420
```

Region: Northwest (1), Northeast (2), Bajío (3), CDMX (4), South/East (5), South (6).

[7]:
```
pd.DataFrame(data1['REGION'].value_counts()).sort_values('REGION').head(6)
```

[7]:
```
        count
REGION
1        2431
2        2499
3        2581
4         952
5        2528
6        2511
```

Education: ranging from 0 (No studies) to 11 (PhD), and 99 for anyone who answered “Doesn’t know”.

[8]:
```
pd.DataFrame(data1['NIV'].value_counts()).sort_values('NIV').head(13)
```

[8]:
```
     count
NIV
```

```
0       532
1        20
2      2697
3      3635
4        15
5       261
6      2808
7       324
8      2853
9        52
10      256
11       44
99        5
```

Each question from this Subsection can be aswered with a number: "Agree" (1), "Doesn't agree nor disagree" (2), "Disagrees" (3), "Doesn't answer" (8) o "Doesn't know" (9).

[9]:
```
### Number of question: 1 to 6

numq = 1

pd.DataFrame(data1[f'P4_{numss}_{numq}'].value_counts()).
 ↪sort_values(f'P4_{numss}_{numq}').head(5)
```

[9]:
```
        count
P4_8_1
1        5515
2        3121
3        4722
8          51
9          93
```

Back to the Intro

1.5 Data normalization and KPIs

Now we define auxiliary columns as functions of the original entries, either to aggregate data, to help with visualizations, of to define useful emtrics.

[10]:
```
### Sex, alphanumeric

def set_sexo(a):
    if a == 1:
        return 'M'
    elif a == 2:
        return 'F'
    else:
        return "N/A"
```

```python
### Region alphanumeric

def set_region(a):
    if a == 1:
        return 'Northwest'
    elif a == 2:
        return 'Northeast'
    elif a == 3:
        return 'Bajío'
    elif a == 4:
        return 'CDMX'
    elif a == 5:
        return 'South/East'
    elif a == 6:
        return 'South'

### Normalized answers in {1,...,5}

def set_norm(a):
    if int(a) == 8:
        return 4
    elif int(a) == 9:
        return 5
    else:
        return int(a)

### Answers, alphanumeric

def set_norm_t(a):
    if int(a) == 1:
        return "Agrees"
    elif int(a) == 2:
        return "Doesn't agree nor disagree"
    elif int(a) == 3:
        return "Disagrees"
    elif int(a) == 4:
        return "Doesn't answer"
    elif int(a) == 5:
        return "Doesn't know"

### Grading: 'Agrees' corresponds to +1, 'Never' corresponds to -1, and
 ↪'Doesn't agree nor disagree' corresponds to 0

def set_norm_calif(a):

    return 2-a
```

```
### Total grade: -6 pts corresponds to 5, and 6 pts corresponds to 10

def set_tot_calif(a):

    return 5/12*(a-6)+10

### Education, alphanumeric, aggregated (for visualizations)

def set_niv(a):
    if int(a) == 0:
        return "No studies"
    elif int(a) < 6:
        return "Basic"
    elif int(a) < 8:
        return "high School"
    elif int(a) == 8:
        return "Undergraduate"
    elif int(a) < 99:
        return "Graduate"
    else:
        return "Doesn't know"
```

New columns are added.

[11]:
```
### Alphanumeric sex

data1.loc[:,'SEXO_HM'] = data1['SEXO'].map(set_sexo)
```

```
C:\Users\patju\AppData\Local\Temp\ipykernel_5664\3748415747.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data1.loc[:,'SEXO_HM'] = data1['SEXO'].map(set_sexo)
```

[12]:
```
### Alphanumeric region

data1.loc[:,'REGION_T'] = data1['REGION'].map(set_region)
```

```
C:\Users\patju\AppData\Local\Temp\ipykernel_5664\821602288.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data1.loc[:,'REGION_T'] = data1['REGION'].map(set_region)
```

[13]:
```python
### Alphanumeric education

data1.loc[:,'NIVED_T'] = data1['NIV'].map(set_niv)
```

```
C:\Users\patju\AppData\Local\Temp\ipykernel_5664\3861912701.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data1.loc[:,'NIVED_T'] = data1['NIV'].map(set_niv)
```

Normalized and alphanumeric answers.

[14]:
```python
for numq in range(1,7):
    data1.loc[:,f'p4{numss}{numq}'] = data1[f'P4_{numss}_{numq}'].map(set_norm)
    data1.loc[:,f'p4{numss}{numq}_t'] = data1[f'p4{numss}{numq}'].
  ↪map(set_norm_t)
    data1 = data1.drop(f'P4_{numss}_{numq}',axis=1)
```

```
C:\Users\patju\AppData\Local\Temp\ipykernel_5664\3780884663.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data1.loc[:,f'p4{numss}{numq}'] = data1[f'P4_{numss}_{numq}'].map(set_norm)
C:\Users\patju\AppData\Local\Temp\ipykernel_5664\3780884663.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data1.loc[:,f'p4{numss}{numq}_t'] = data1[f'p4{numss}{numq}'].map(set_norm_t)
```

We get our new, provisional table. Later, after filtering, we will add a KPI column.

[15]:
```python
data1 = data1[['LLAVEMOD', 'SEXO', 'SEXO_HM', 'EDAD_V', 'NIV','NIVED_T',␣
  ↪'REGION',␣
  ↪'REGION_T','TLOC',f'p4{numss}1',f'p4{numss}2',f'p4{numss}3',f'p4{numss}4',f'p4{numss}5',f'p
data1.head()
```

[15]:
```
    LLAVEMOD  SEXO SEXO_HM  EDAD_V  NIV        NIVED_T  REGION   REGION_T  \
0     101101     1       M      38    8  Undergraduate       1  Northwest
1     210101     1       M      36    8  Undergraduate       6      South
2     303102     1       M      20    3          Basic       4       CDMX
```

```
3    401101      1        M      59     3          Basic        1  Northwest
4    503101      2        F      37     6    high School        4       CDMX

   TLOC  p481  …  p483  p484  p485  p486                          p481_t  \
0     1     1  …     1     1     1     2                          Agrees
1     3     3  …     1     1     1     2                       Disagrees
2     1     2  …     3     2     1     1  Doen't agree nor disagree
3     1     1  …     2     2     1     1                          Agrees
4     1     3  …     2     1     3     3                       Disagrees

                      p482_t                     p483_t  \
0                     Agrees                     Agrees
1  Doen't agree nor disagree                     Agrees
2                  Disagrees                  Disagrees
3                     Agrees  Doen't agree nor disagree
4                  Disagrees  Doen't agree nor disagree

                      p484_t     p485_t                     p486_t
0                     Agrees     Agrees  Doen't agree nor disagree
1                     Agrees     Agrees  Doen't agree nor disagree
2  Doen't agree nor disagree     Agrees                     Agrees
3  Doen't agree nor disagree     Agrees                     Agrees
4                     Agrees  Disagrees                  Disagrees

[5 rows x 21 columns]
```

Back to the Intro

1.6 Distributions

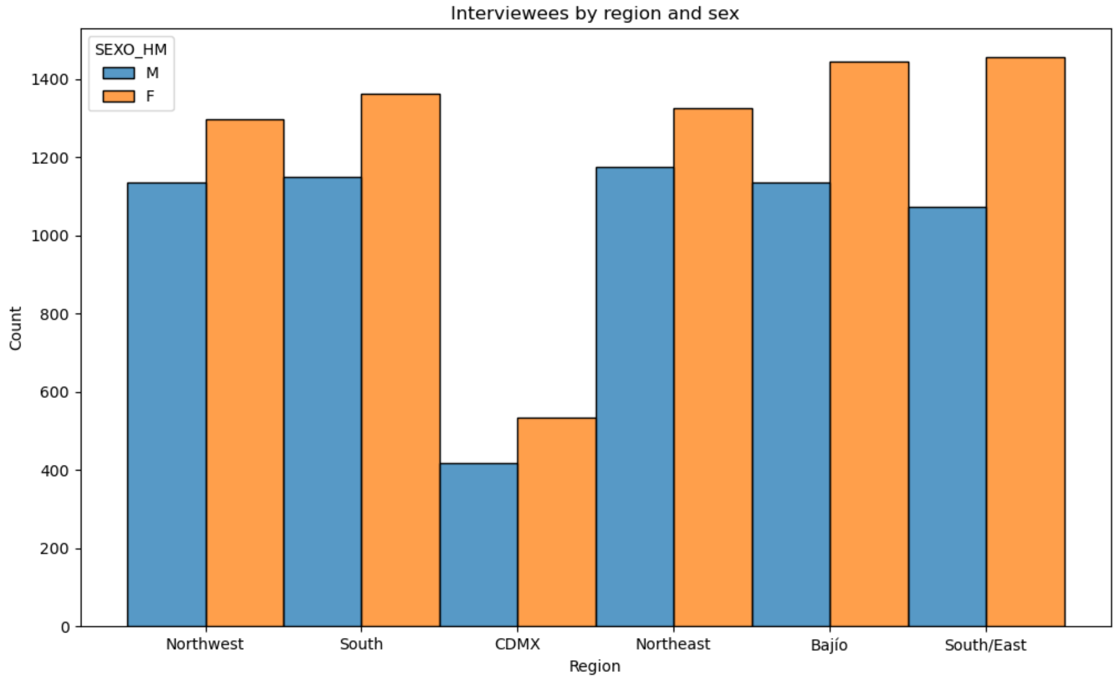
Before beginning with the predictive analysis, we will visualize our data to better understand it.

1.6.1 Overall distro

We plot sex, region, and education.

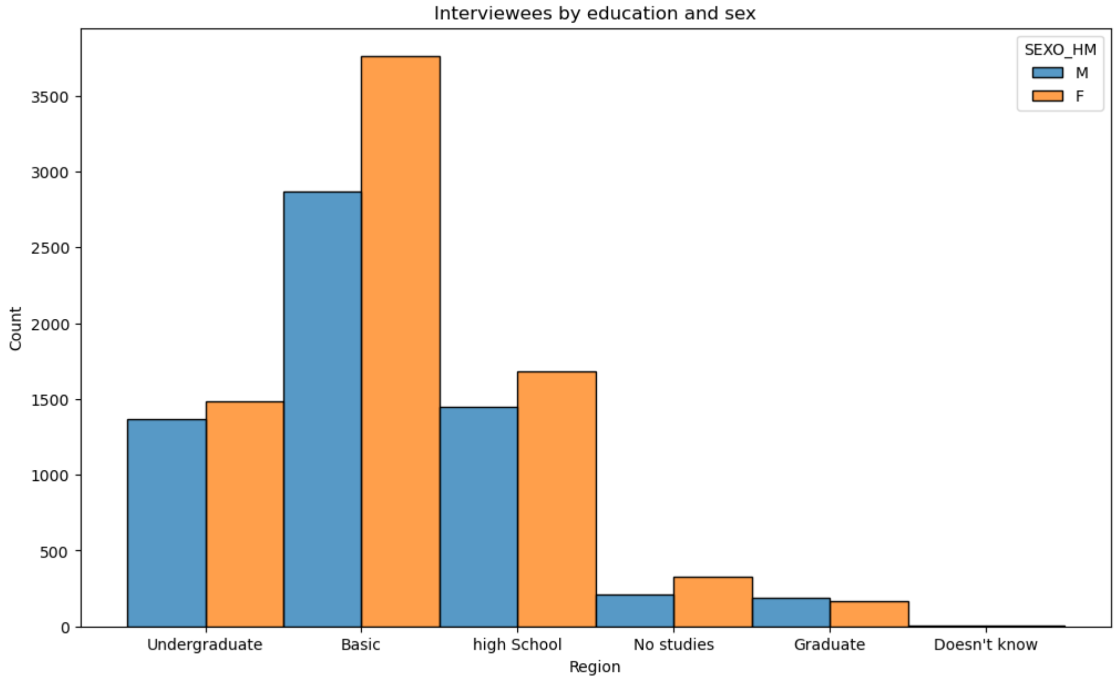
[16]:
```
fig, ax = plt.subplots(figsize=(12,7))

plot1 = sns.histplot(data1, x = 'REGION_T', multiple="dodge", hue = 'SEXO_HM')

ax.set_xlabel('Region')
ax.set_ylabel('Count')
ax.set_title('Interviewees by region and sex')
```

[16]:
```
Text(0.5, 1.0, 'Interviewees by region and sex')
```



[65]:
```
fig, ax = plt.subplots(figsize=(12,7))

plot1 = sns.histplot(data1, x = 'NIVED_T', multiple="dodge", hue = 'SEXO_HM')

ax.set_xlabel('Region')
ax.set_ylabel('Count')
ax.set_title('Interviewees by education and sex')
```

[65]: Text(0.5, 1.0, 'Interviewees by education and sex')



1.6.2 Answers

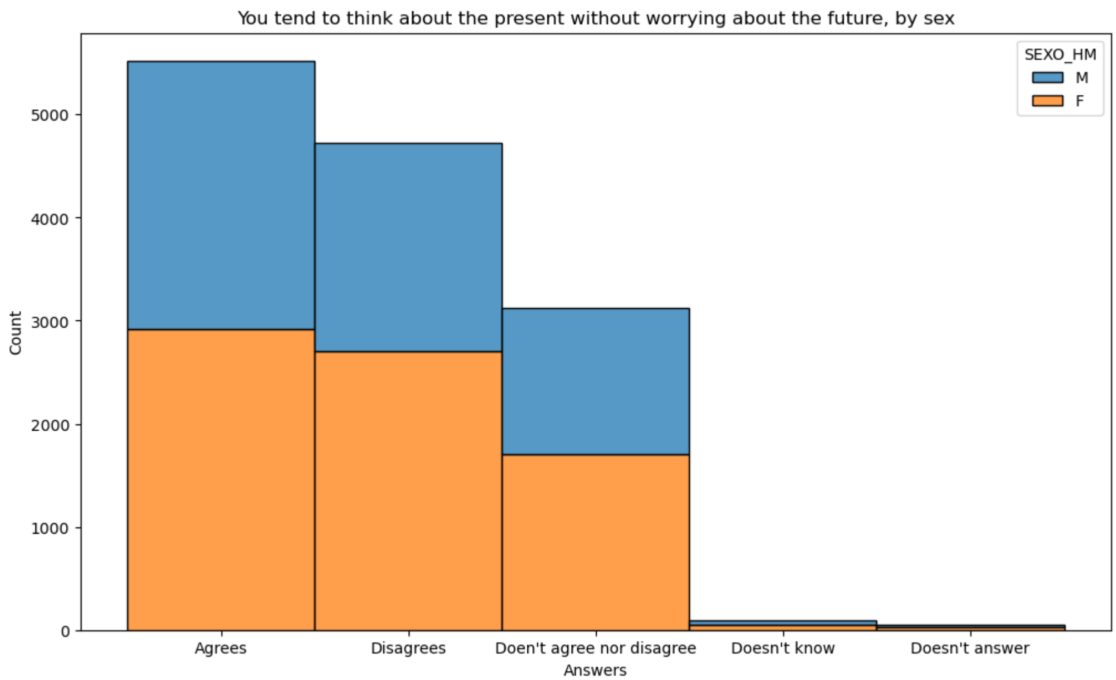
We plot the distributions of the answers with respect to sex, region, and education.

[18]:
```
### Questions dictionnary

questions = {'1' : 'You tend to think about the present without worrying about␣
↪the future',
             '2' : 'Money exists to be spent',
             '3' : 'You keep a detailed attention to your expenditures',
             '4' : 'Given your economic situation, you feel youll get the␣
↪things you want',
             '5' : 'Your money is enough to cover for your expenditures',
             '6' : 'You feel at ease about the money you get being enough'}
```

[19]:
```
### Question

numq = 1

fig, ax = plt.subplots(figsize=(12,7))

### General histogram by sex

plot1 = sns.histplot(data1, x = f'p4{numss}{numq}_t', multiple="stack", hue =␣
↪"SEXO_HM")
```

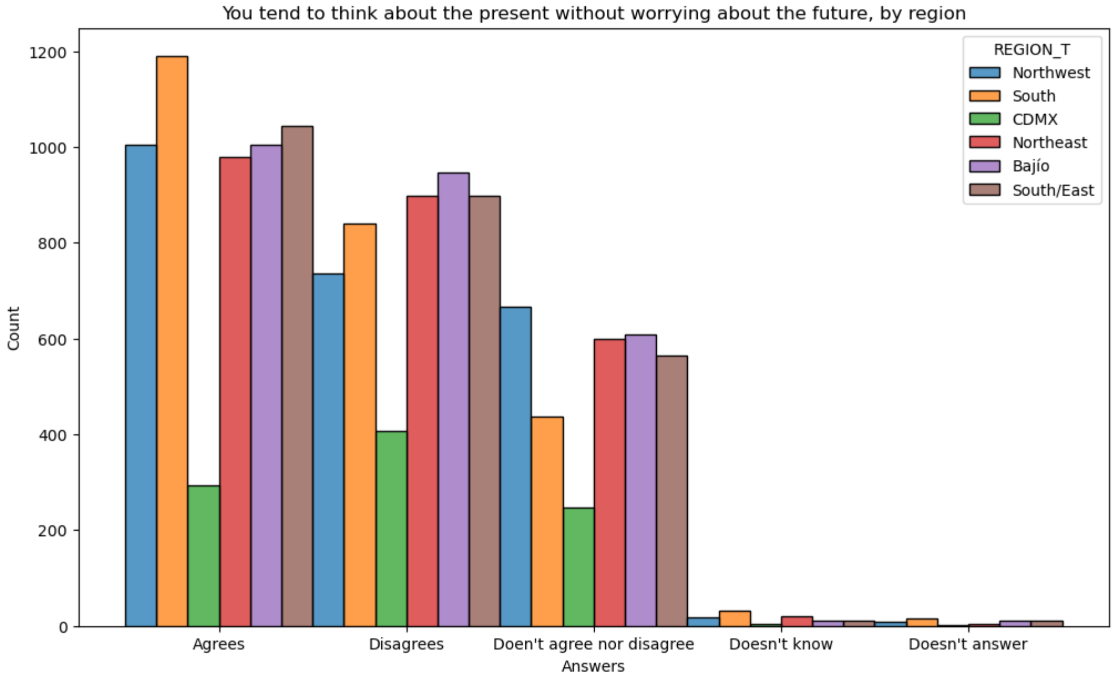
```
ax.set_xlabel('Answers')
ax.set_ylabel('Count')
ax.set_title(f'{questions[str(numq)]}, by sex')
```

[19]: Text(0.5, 1.0, 'You tend to think about the present without worrying about the future, by sex')



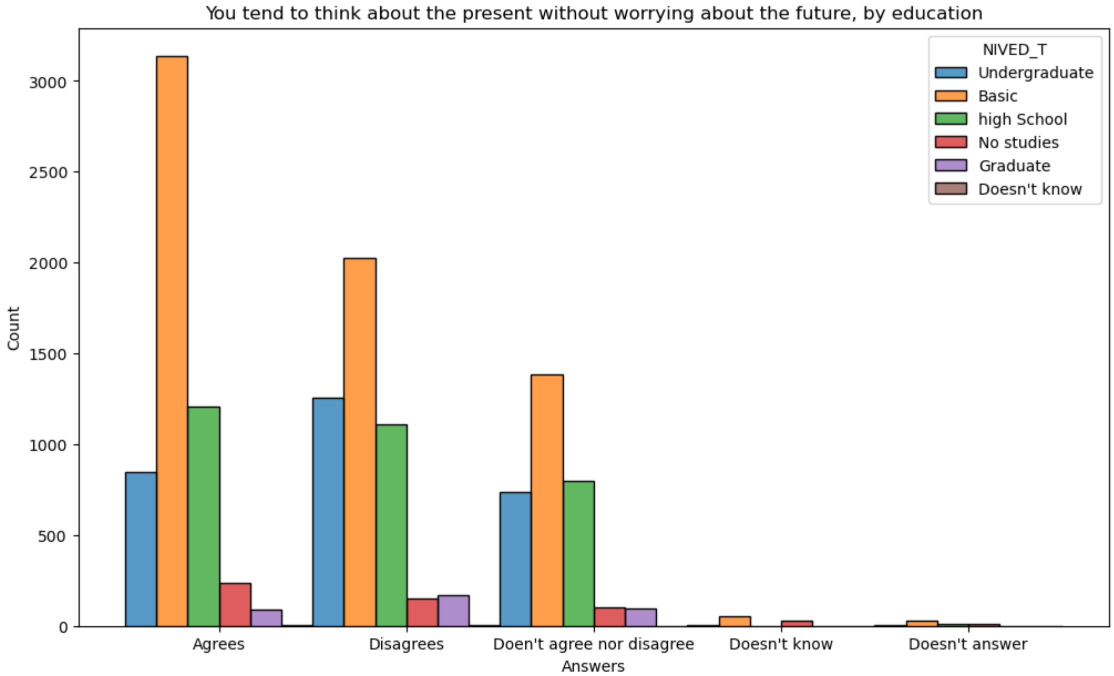
[23]:
```
### Question

numq = 1

fig, ax = plt.subplots(figsize=(12,7))

### General histogram by region

plot1 = sns.histplot(data1, x = f'p4{numss}{numq}_t', multiple="dodge", hue =
↪"REGION_T")

ax.set_xlabel('Answers')
ax.set_ylabel('Count')
ax.set_title(f'{questions[str(numq)]}, by region')
```

[23]: Text(0.5, 1.0, 'You tend to think about the present without worrying about the future, by region')



[24]:
```
### Question

numq = 1

fig, ax = plt.subplots(figsize=(12,7))

### General histogram by region

plot1 = sns.histplot(data1, x = f'p4{numss}{numq}_t', multiple="dodge", hue =␣
 ↪"NIVED_T")

ax.set_xlabel('Answers')
ax.set_ylabel('Count')
ax.set_title(f'{questions[str(numq)]}, by education')
```

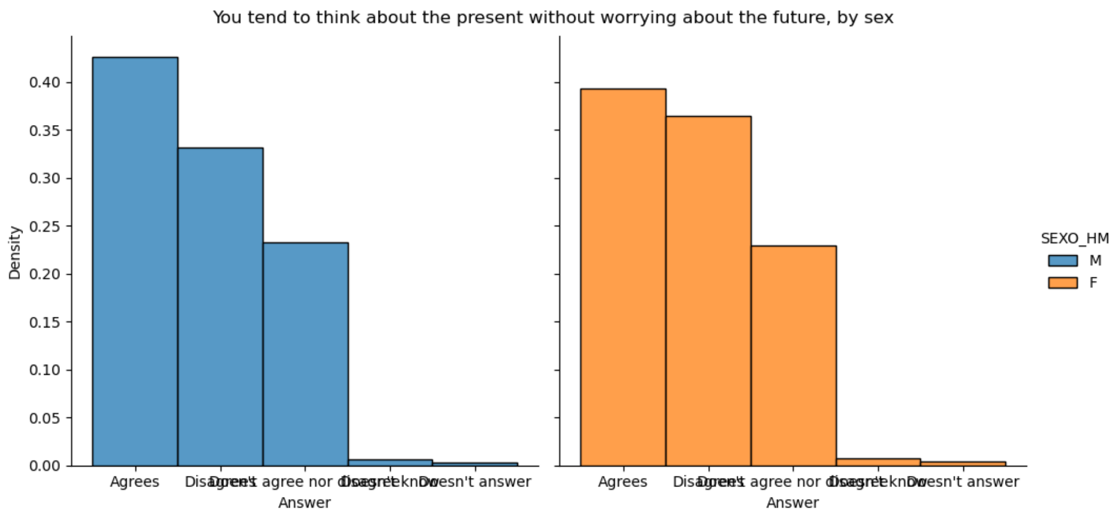
[24]:
```
Text(0.5, 1.0, 'You tend to think about the present without worrying about the
future, by education')
```



We see that some classes, such as females (F) for sex, CDMX for region, and Basic for education, have a significatively different number of answers from the rest of their respective category. To check whether all subpopulations behave similarly, we plot independent histograms.

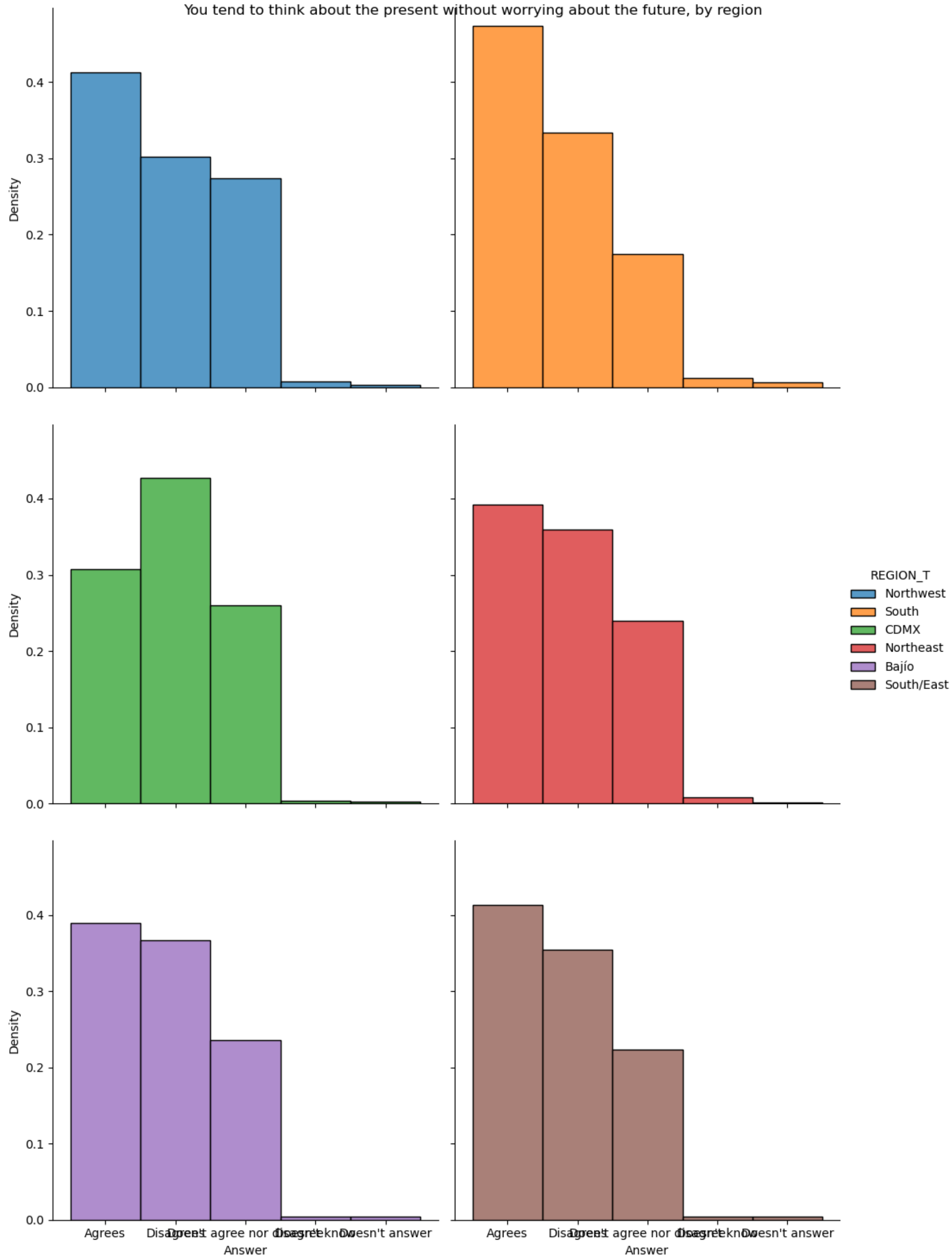
[25]:
```
### Filtered histograms

### Question

numq = 1

### By sex

g = sns.FacetGrid(data1, col="SEXO_HM", hue="SEXO_HM", height = 5)
g.map(sns.histplot, f'p4{numss}{numq}_t', stat='density', bins=10)
g.set_axis_labels("Answer", "Density")
g.set_titles('')
g.fig.suptitle(f'{questions[str(numq)]}, by sex')
g.add_legend()
```

[25]: <seaborn.axisgrid.FacetGrid at 0x20189d181a0>



[29]:
```
### Filtered histograms

### Question

numq = 1

### By region

g = sns.FacetGrid(data1, col="REGION_T", hue="REGION_T", height = 5, col_wrap =␣
↪2)
g.map(sns.histplot, f'p4{numss}{numq}_t', stat='density', bins=10)
g.set_axis_labels("Answer", "Density")
g.set_titles('')
g.fig.suptitle(f'{questions[str(numq)]}, by region')
g.add_legend()
```

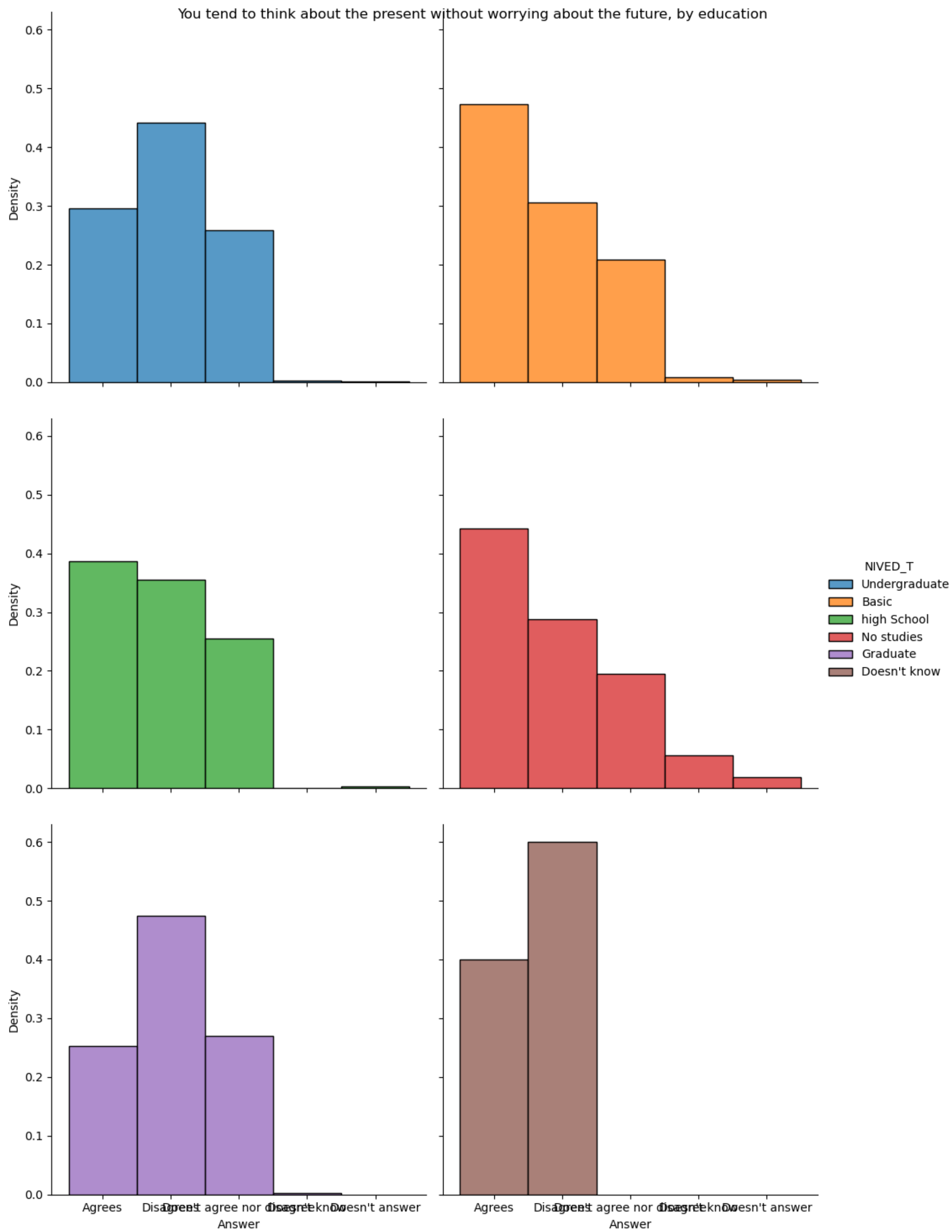
[29]:
```
<seaborn.axisgrid.FacetGrid at 0x20189df12e0>
```



[30]:
```
### Filtered histograms

### Question
```

```python
numq = 1

### By education

g = sns.FacetGrid(data1, col="NIVED_T", hue="NIVED_T", height = 5, col_wrap = 2)
g.map(sns.histplot, f'p4{numss}{numq}_t', stat='density', bins=10)
g.set_axis_labels("Answer", "Density")
g.set_titles('')
g.fig.suptitle(f'{questions[str(numq)]}, by education')
g.add_legend()
```

[30]: <seaborn.axisgrid.FacetGrid at 0x20189f36780>



We see different behaviours amongst subpopulations, more concretely amongst regions and education. We expect this to be addressed by the correlation analysis.

Back to the intro

1.7 Data filtering

Given that some answers were given by very few interviewees, we filter the data to avoid skewing the results.

[31]:
```
data2 = data1
```

[32]:
```
### 'Doesnt know' or 'Doesnt answer' responses are filtered out

for numq in range(1,7):

    data2 = data2.loc[data1[f'p4{numss}{numq}']<4]

data2 = data2.loc[data1['NIVED_T']!="Doesn't know"]
```

[33]:
```
### Dimensiones

data2 = data2.dropna()
data2.shape
```

[33]:
```
(13114, 21)
```

[34]:
```
print(f'After filtering "Doesnt answer" and "Doesnt know", we get {round(data2.
↪shape[0]/data1.shape[0]*100,2)}% of the original sample')
print(f'{round((data1.shape[0]-data2.shape[0])/data1.shape[0]*100,2)}% of the␣
↪original sample was filtered out')
```

```
After filtering "Doesnt answer" and "Doesnt know", we get 97.13% of the original
sample
2.87% of the original sample was filtered out
```

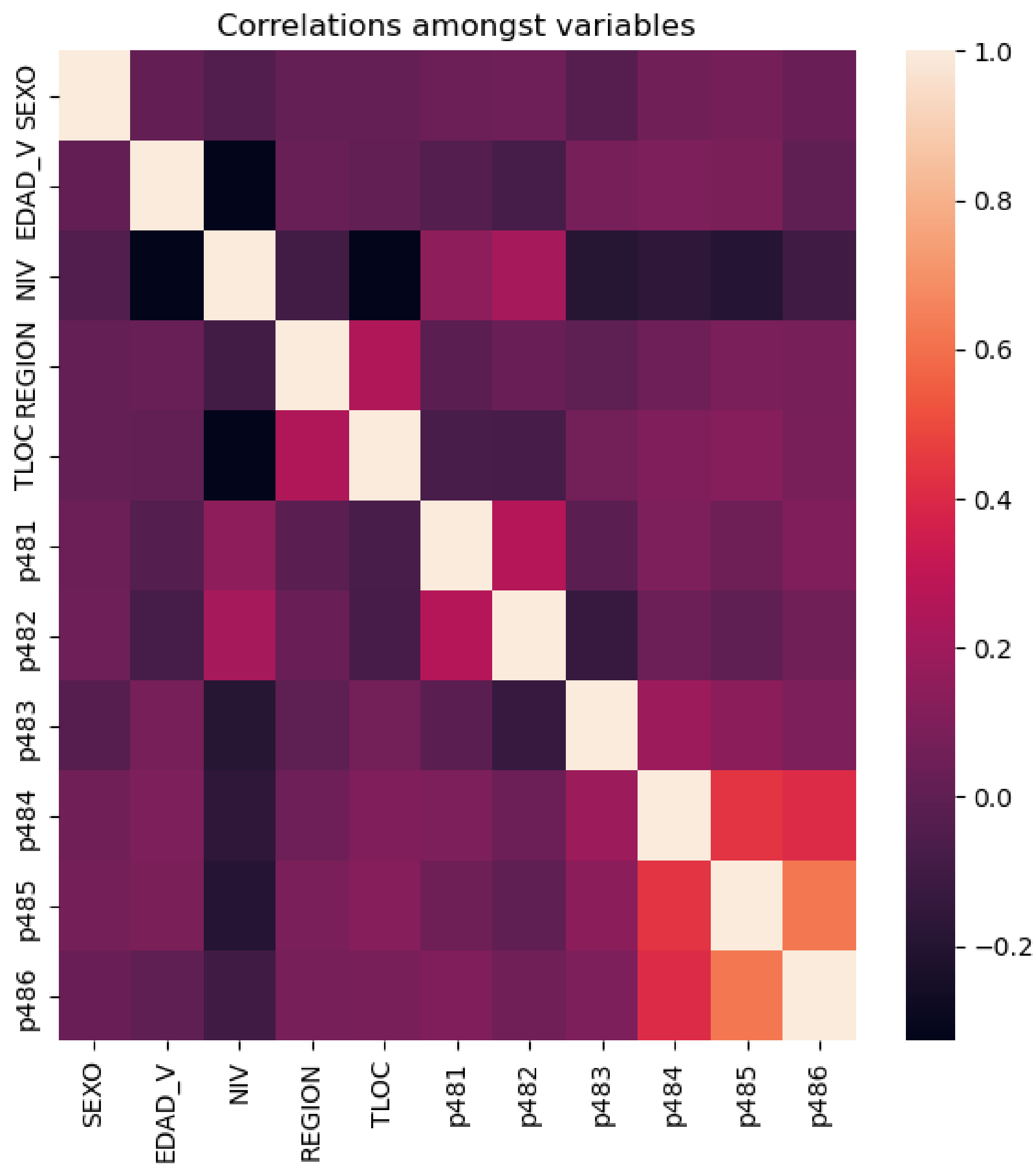
Back to the Intro

1.8 Correlation analysis

We wish to establish some relationships amongst our data prior to conducting the predictive analysis.

[35]:
```
corr = data2.drop(['LLAVEMOD'],axis=1).select_dtypes('number').corr()

fig, ax = plt.subplots(figsize=(7,7))

# plot the heatmap
sns.heatmap(corr)

ax.set_title('Correlations amongst variables')
```

[35]:
```
Text(0.5, 1.0, 'Correlations amongst variables')
```



[36]:

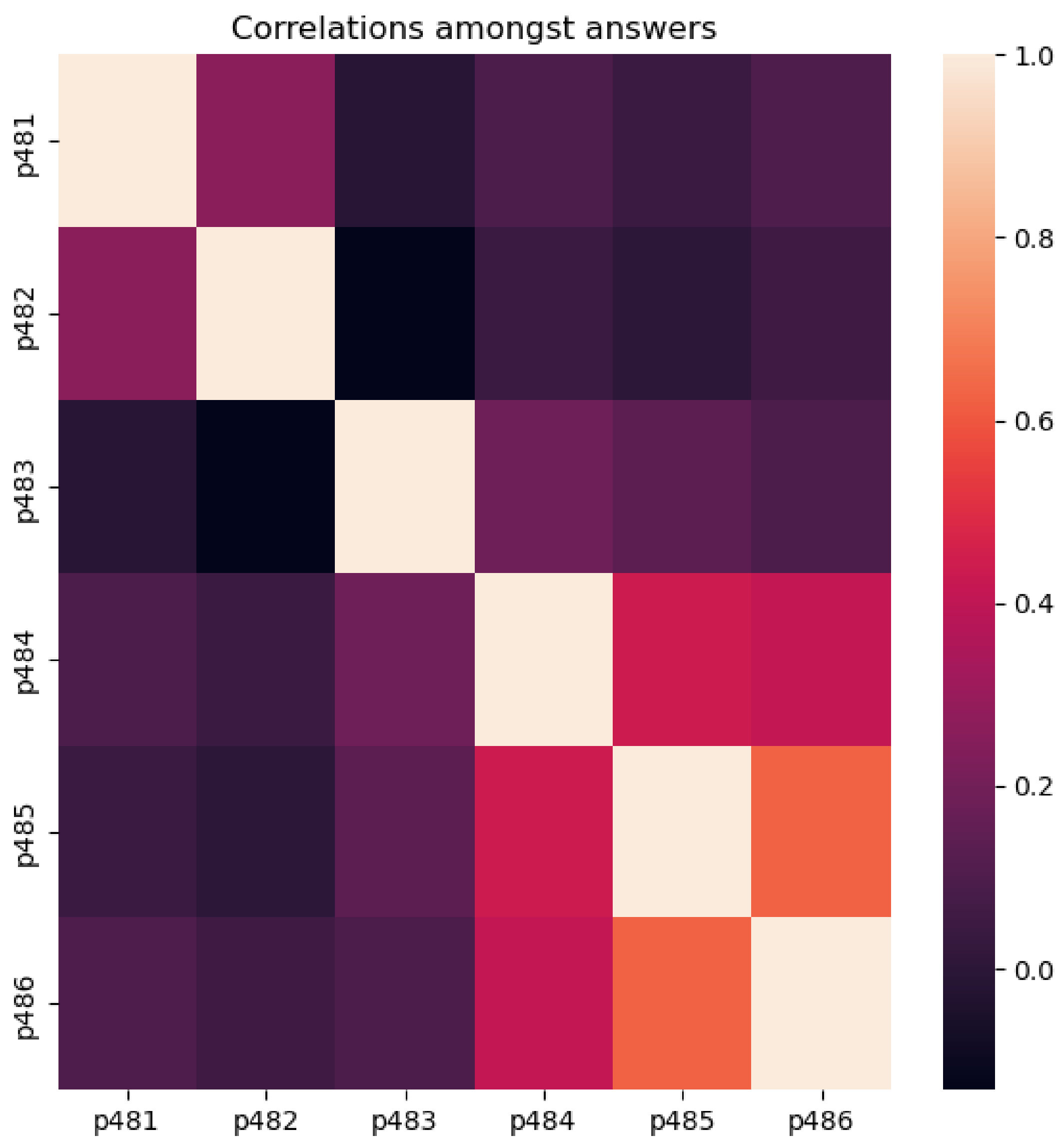
```
corr[[f'p4{numss}1',f'p4{numss}2',f'p4{numss}3',f'p4{numss}4',f'p4{numss}5',f'p4{numss}6']].
 ↪head(5)
```

[36]:

	p481	p482	p483	p484	p485	p486
SEXO	0.039200	0.048617	-0.029543	0.057105	0.071575	0.033751
EDAD_V	-0.032001	-0.082906	0.076390	0.093111	0.089345	0.001494
NIV	0.149092	0.220770	-0.188372	-0.161685	-0.192363	-0.099845
REGION	-0.012665	0.032984	-0.002348	0.051212	0.084963	0.075229
TLOC	-0.072602	-0.077082	0.065300	0.099339	0.123469	0.078961

[37]:
```
corr1 = data2.drop(['LLAVEMOD','REGION','SEXO','EDAD_V', 'NIV','TLOC'],axis=1).
 ↪select_dtypes('number').corr()

fig, ax = plt.subplots(figsize=(7,7))

# plot the heatmap
sns.heatmap(corr1)

ax.set_title('Correlations amongst answers')
```

[37]:
```
Text(0.5, 1.0, 'Correlations amongst answers')
```



[38]:
```
corr1.head(6)
```

[38]:
```
          p481      p482      p483      p484      p485      p486
p481  1.000000  0.260251 -0.011931  0.093250  0.041523  0.099546
p482  0.260251  1.000000 -0.133056  0.039868  0.003067  0.056486
p483 -0.011931 -0.133056  1.000000  0.188933  0.138980  0.093603
p484  0.093250  0.039868  0.188933  1.000000  0.436564  0.407321
p485  0.041523  0.003067  0.138980  0.436564  1.000000  0.624730
p486  0.099546  0.056486  0.093603  0.407321  0.624730  1.000000
```

We see an important correlation cluster between question 4 (getting the things you want), 5 (covering expenditures) and 6 (feeling at ease). There seems to be no significative correlation with sex, age, education or locality variables, thus, there is no one evident factor to answering each of the questions.

Back to the Intro

1.9 Key indicator

Now we define a metric that determines the likelihood of contracting debt. We take "Agrees" as +1, "Disagrees" as -1, and "Doesn't agree nor disagree" as 0. Then we sum these independent scores to get a grade ranging from -6 to 6; finally, this grading is transformed to a final score ranging from 5 to 10, where higher scores indicate more economic and stability and a fewer probability of contracting debt. Note that the answers to questions 1 and 2 must be reversed in order to be consistent with the previous interpretation.

[39]:
```
for numq in range(1,7):
    data2[f'c4{numss}{numq}'] = data1[f'p4{numss}{numq}'].map(set_norm_calif)

data2[f't4{numss}'] =␣
 ↪data2[f'c4{numss}3']+data2[f'c4{numss}4']-data2[f'c4{numss}1']-data2[f'c4{numss}2']+data2[f

data2[f't4{numss}_calif'] = data2[f't4{numss}'].map(set_tot_calif)
```

[40]:
```
data3 = data2

for numq in range(1,7):
    data3 = data3.drop(f'c4{numss}{numq}', axis = 1)

data3 = data3.drop(f't4{numss}', axis = 1)
data3 = data3.dropna()
data3.head()
```

[40]:
```
   LLAVEMOD  SEXO SEXO_HM  EDAD_V  NIV        NIVED_T  REGION   REGION_T  \
0    101101     1       M      38    8  Undergraduate       1  Northwest
1    210101     1       M      36    8  Undergraduate       6      South
2    303102     1       M      20    3          Basic       4       CDMX
3    401101     1       M      59    3          Basic       1  Northwest
4    503101     2       F      37    6    high School       4       CDMX
```

```
   TLOC  p481  ...  p484  p485  p486                        p481_t  \
0     1     1  ...     1     1     2                        Agrees
1     3     3  ...     1     1     2                     Disagrees
2     1     2  ...     2     1     1  Doen't agree nor disagree
3     1     1  ...     2     1     1                        Agrees
4     1     3  ...     1     3     3                     Disagrees

                      p482_t                     p483_t  \
0                     Agrees                     Agrees
1  Doen't agree nor disagree                     Agrees
2                  Disagrees                  Disagrees
3                     Agrees  Doen't agree nor disagree
4                  Disagrees  Doen't agree nor disagree

                      p484_t     p485_t                     p486_t t48_calif
0                     Agrees     Agrees  Doen't agree nor disagree  7.916667
1                     Agrees     Agrees  Doen't agree nor disagree  9.166667
2  Doen't agree nor disagree     Agrees                     Agrees  8.333333
3  Doen't agree nor disagree     Agrees                     Agrees  7.500000
4                     Agrees  Disagrees                  Disagrees  7.916667

[5 rows x 22 columns]
```

In the sequel, we will work with table `data3` for visualizing the KPI and conducting the predictive analysis. We can export this data to other visualizing tools.

[38]:
```
data3.to_csv(f'./tabla_4{numss}.csv')
```

1.9.1 Visualizing the KPI

We analyze the distribution of the KPI with respect to the independent variables. As it is a continuous metric, we start with simple descriptive statistics.

[41]:
```
pd.DataFrame(data3[f't4{numss}_calif']).describe()
```

[41]:
```
          t48_calif
count  13114.000000
mean       7.614032
std        1.097954
min        5.000000
25%        6.666667
50%        7.500000
75%        8.333333
max       10.000000
```

We obtain the mean $\bar{x} = 7.61$, the median $\mathrm{MED} = 7.5$, and the standard deviation $\sigma = 1.09$. The mean is slightly greater than the median, which indicates that the values in the third and fourth quartiles are sparser.

[42]:
```python
mu = 7.614
sig = 1.0979

print(f'We claim that 95% of the sample is contained within the range␣
↪[{round(mu-2*sig,2)},{round(mu+2*sig,2)}].')
```

```
We claim that 95% of the sample is contained within the range [5.42,9.81].
```

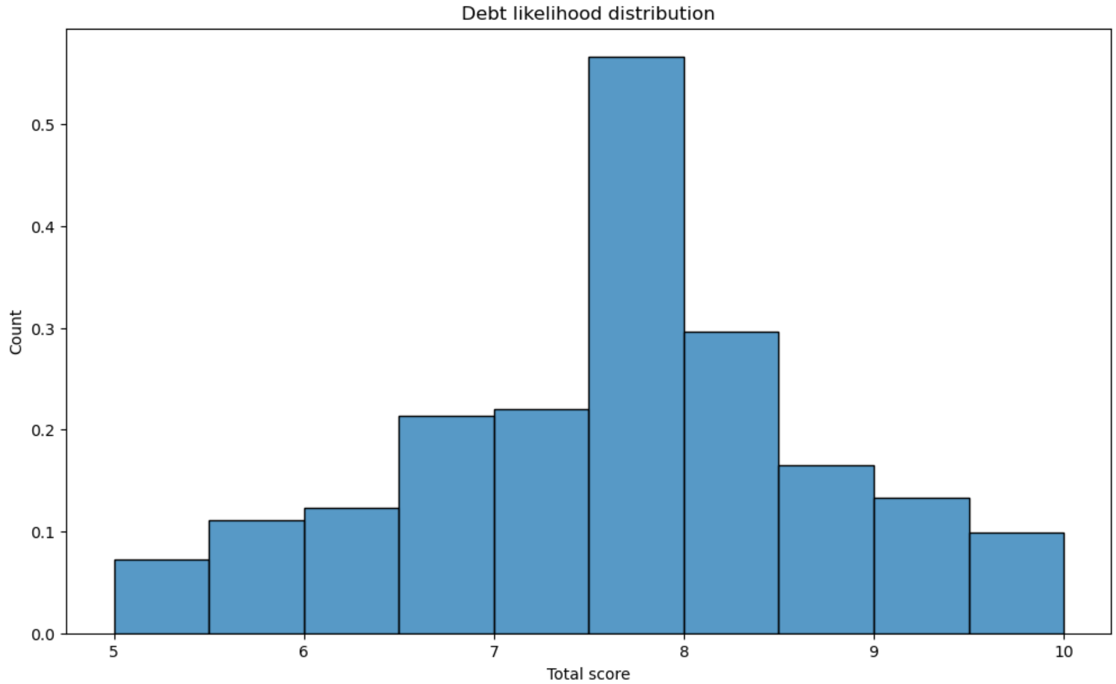
[43]:
```python
q1 = 6.666
q3 = 8.333
k = 1.5

print(f'Moreover, any entry less than {round(q1-k*(q3-q1),2)} and greater than␣
↪{round(q3+k*(q3-q1),2)} is considered as outliers.')
```

```
Moreover, any entry less than 4.17 and greater than 10.83 are considered as
outliers.
```

Now we visualize the KPI.

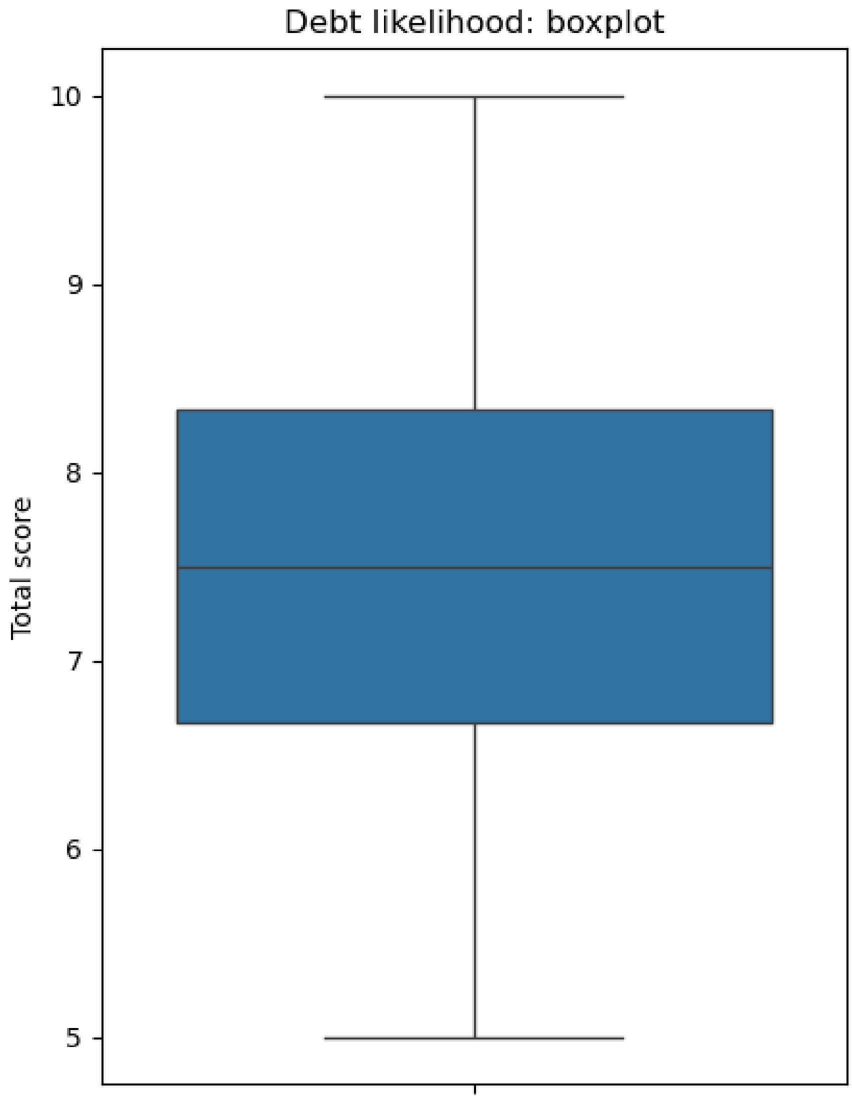
[44]:
```python
### General histogram

fig, ax = plt.subplots(figsize=(12,7))

plot1 = sns.histplot(data3, x = f't4{numss}_calif', stat = 'density', bins=10)

ax.set_xlabel('Total score')
ax.set_ylabel('Count')
ax.set_title('Debt likelihood distribution')
```

[44]:
```
Text(0.5, 1.0, 'Debt likelihood distribution')
```



[45]:
```
### General boxplot

fig, ax = plt.subplots(figsize=(5,7))

plot1 = sns.boxplot(data3, y = f't4{numss}_calif')

ax.set_ylabel('Total score')
ax.set_title('Debt likelihood: boxplot')
```

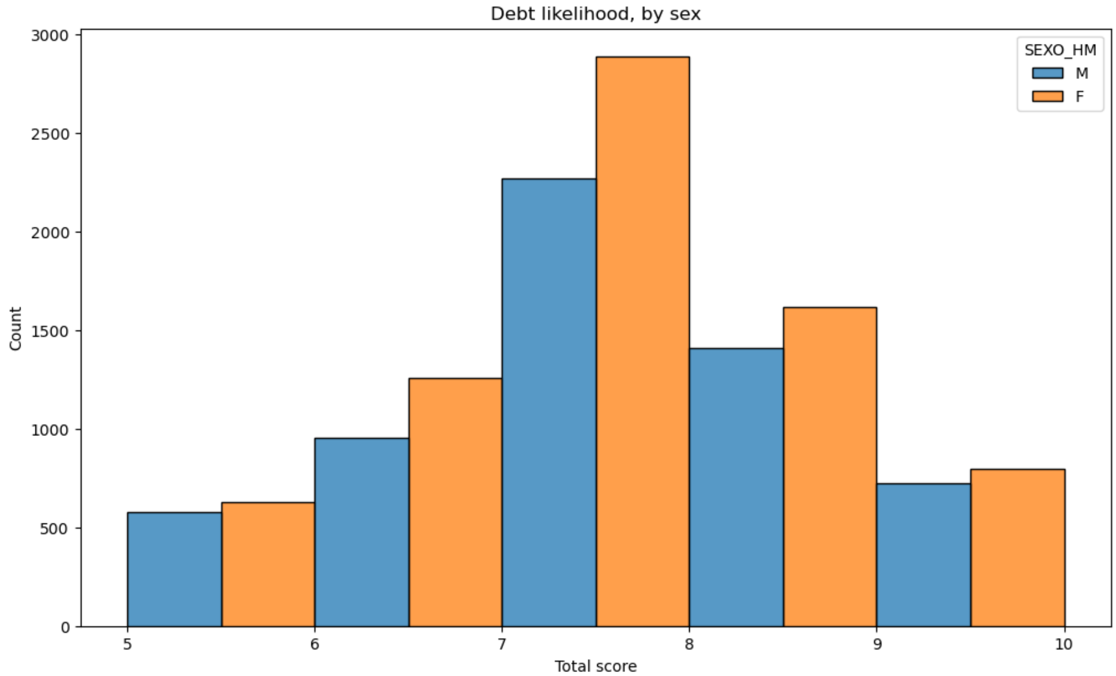
[45]: Text(0.5, 1.0, 'Debt likelihood: boxplot')



KPI by sex

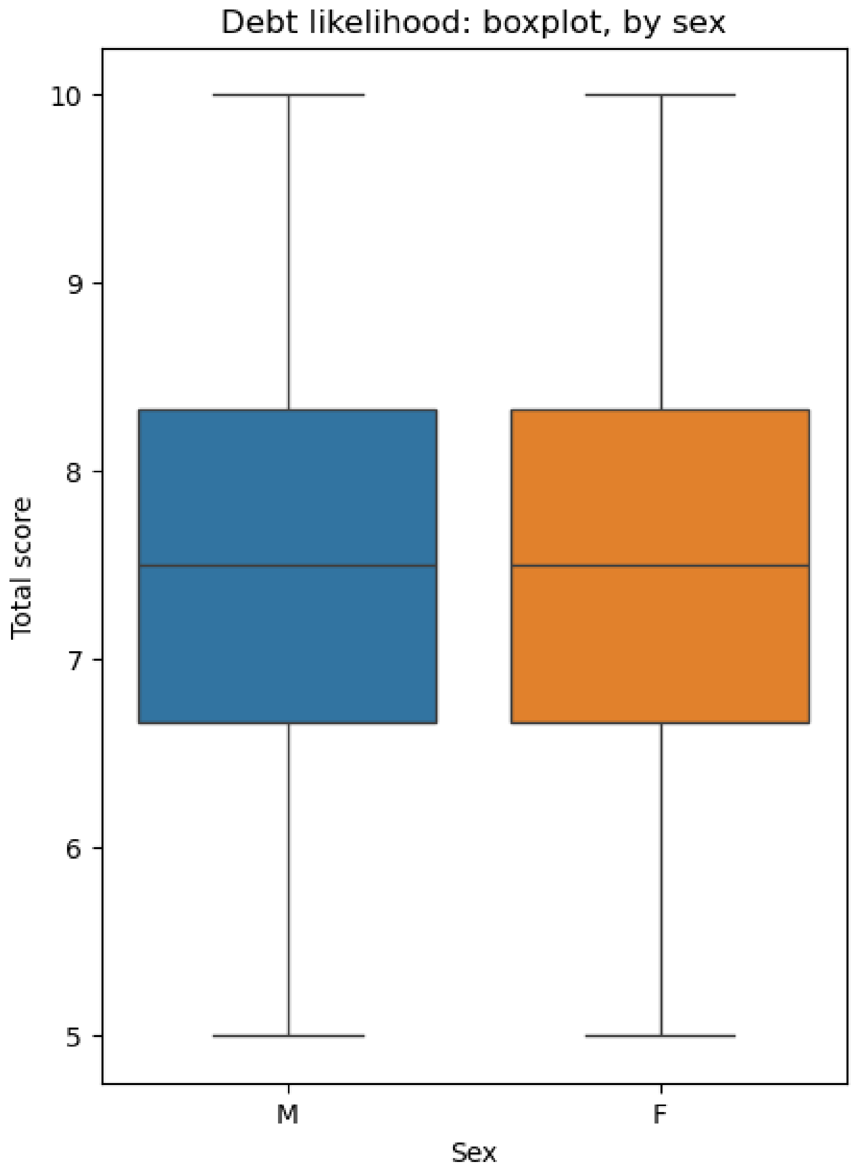
[46]:
```
### Histograms by variables

fig, ax = plt.subplots(figsize=(12,7))

plot1 = sns.histplot(data3, x = f't4{numss}_calif', multiple="dodge", hue =␣
 ↪'SEXO_HM', bins=5)

ax.set_xlabel('Total score')
ax.set_ylabel('Count')
ax.set_title('Debt likelihood, by sex')
```

[46]: Text(0.5, 1.0, 'Debt likelihood, by sex')



[51]:
```
### Boxplot by sex

fig, ax = plt.subplots(figsize=(5,7))

plot1 = sns.boxplot(data3, x = 'SEXO_HM', y = f't4{numss}_calif', hue =␣
 ↪'SEXO_HM')

ax.set_ylabel('Total score')
ax.set_xlabel('Sex')
ax.set_title('Debt likelihood: boxplot, by sex')
```

[51]: Text(0.5, 1.0, 'Debt likelihood: boxplot, by sex')

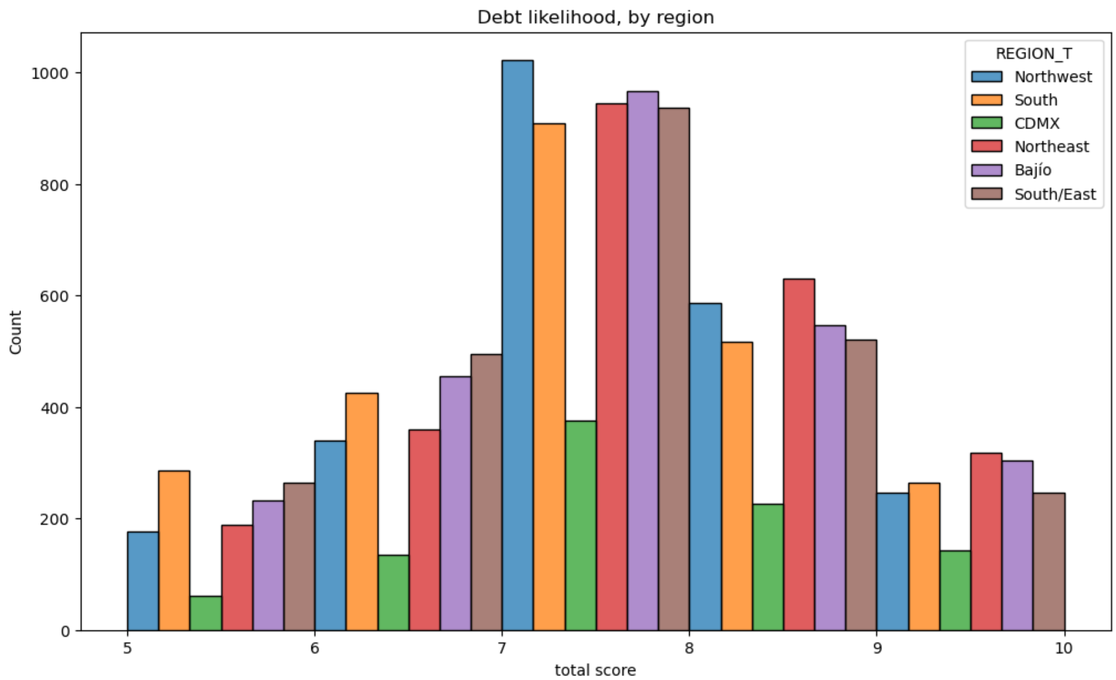


1.9.2 KPI by region

[48]:
```
### Histograms by variables

fig, ax = plt.subplots(figsize=(12,7))

plot1 = sns.histplot(data3, x = f't4{numss}_calif', multiple="dodge", hue =␣
↪'REGION_T', bins=5)

ax.set_xlabel('total score')
ax.set_ylabel('Count')
```

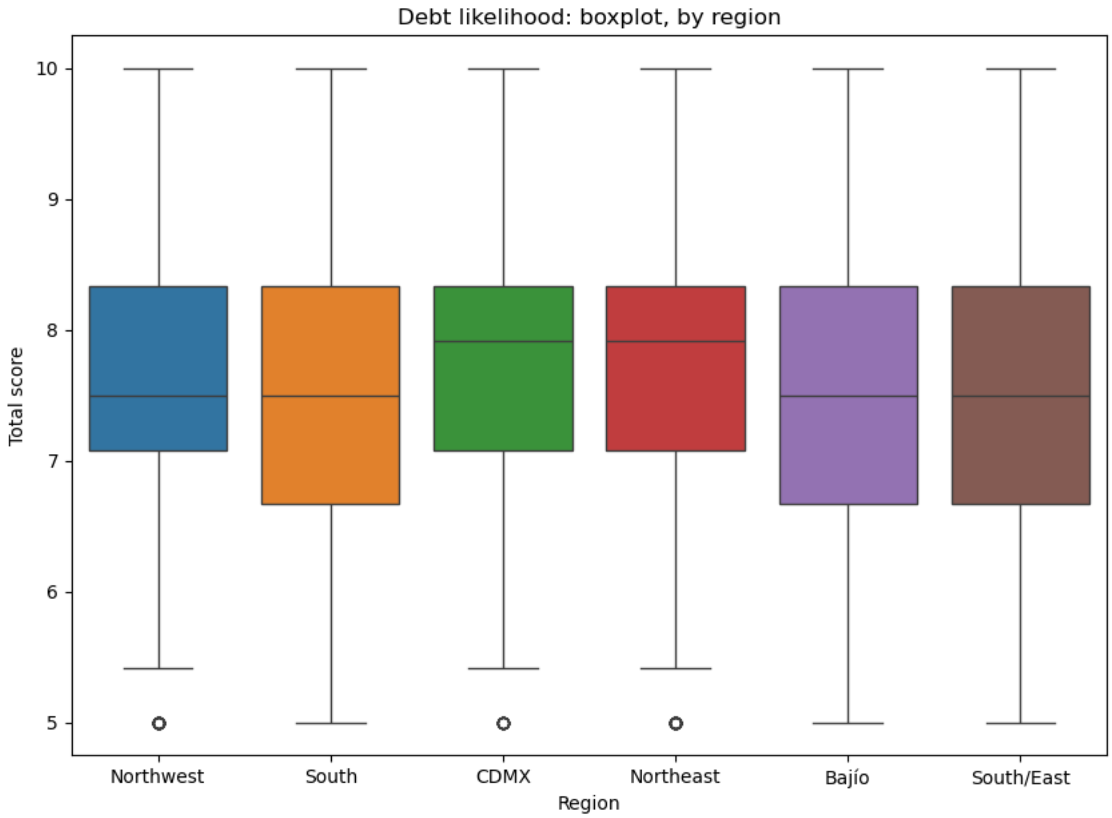
```
ax.set_title('Debt likelihood, by region')
```

[48]: Text(0.5, 1.0, 'Debt likelihood, by region')



[52]:
```
### Boxplot by region

fig, ax = plt.subplots(figsize=(10,7))

plot1 = sns.boxplot(data3, x = 'REGION_T', y = f't4{numss}_calif', hue =␣
↪'REGION_T')

ax.set_ylabel('Total score')
ax.set_xlabel('Region')
ax.set_title('Debt likelihood: boxplot, by region')
```

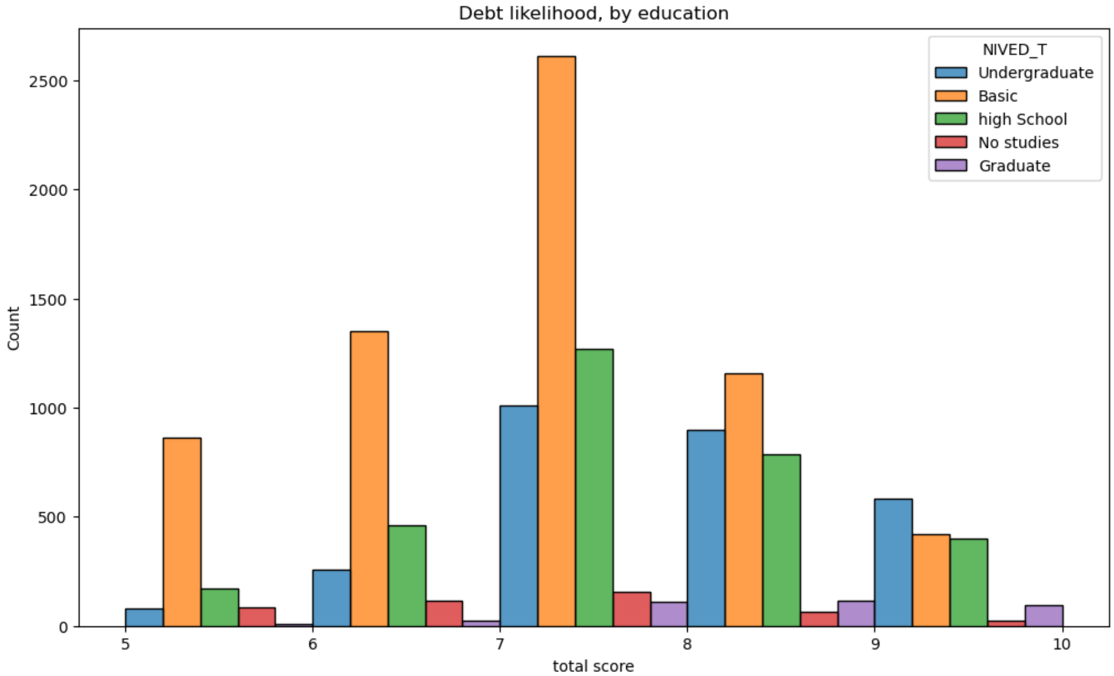
[52]: Text(0.5, 1.0, 'Debt likelihood: boxplot, by region')



1.9.3 KPI by education

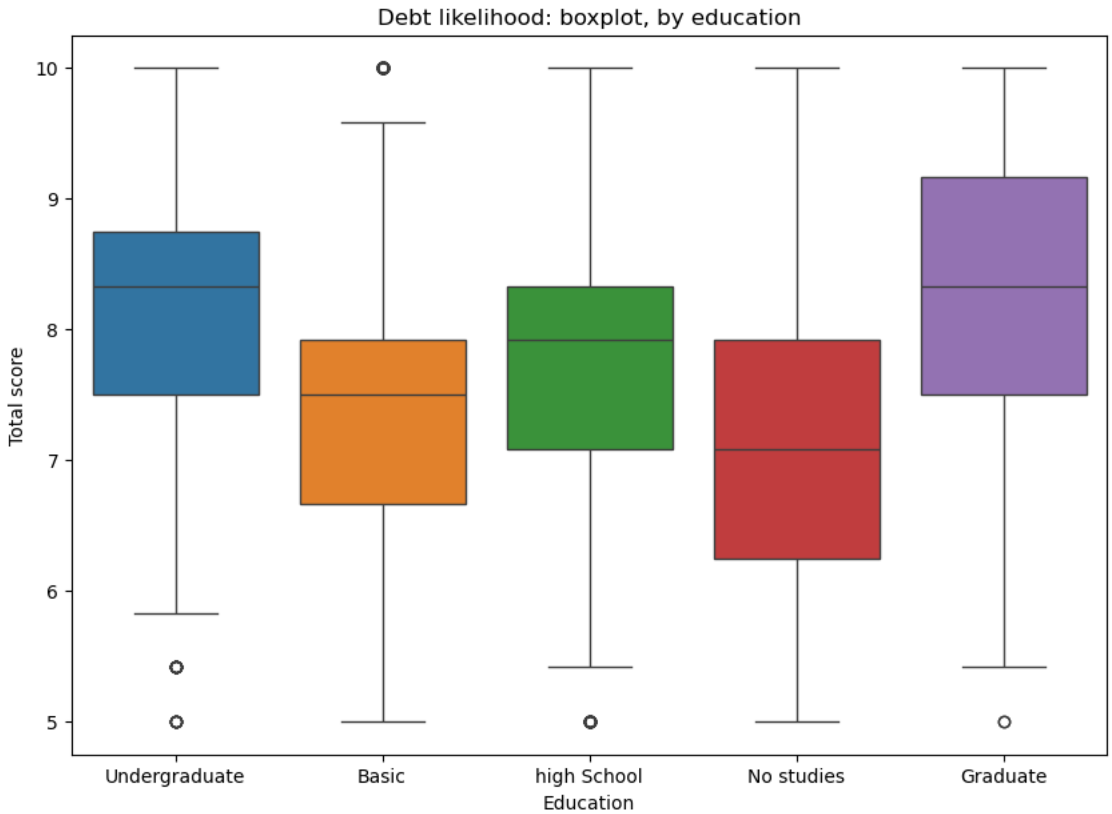
[50]:
```
### Histograms by variable

fig, ax = plt.subplots(figsize=(12,7))

plot1 = sns.histplot(data3, x = f't4{numss}_calif', multiple="dodge", hue =␣
↪'NIVED_T', bins=5)

ax.set_xlabel('total score')
ax.set_ylabel('Count')
ax.set_title('Debt likelihood, by education')
```

[50]: Text(0.5, 1.0, 'Debt likelihood, by education')



[53]:
```
### Boxplot por nivel educativo

fig, ax = plt.subplots(figsize=(10,7))

plot1 = sns.boxplot(data3, x = 'NIVED_T', y = f't4{numss}_calif', hue =␣
    ↪'NIVED_T')

ax.set_ylabel('Total score')
ax.set_xlabel('Education')
ax.set_title('Debt likelihood: boxplot, by education')
```

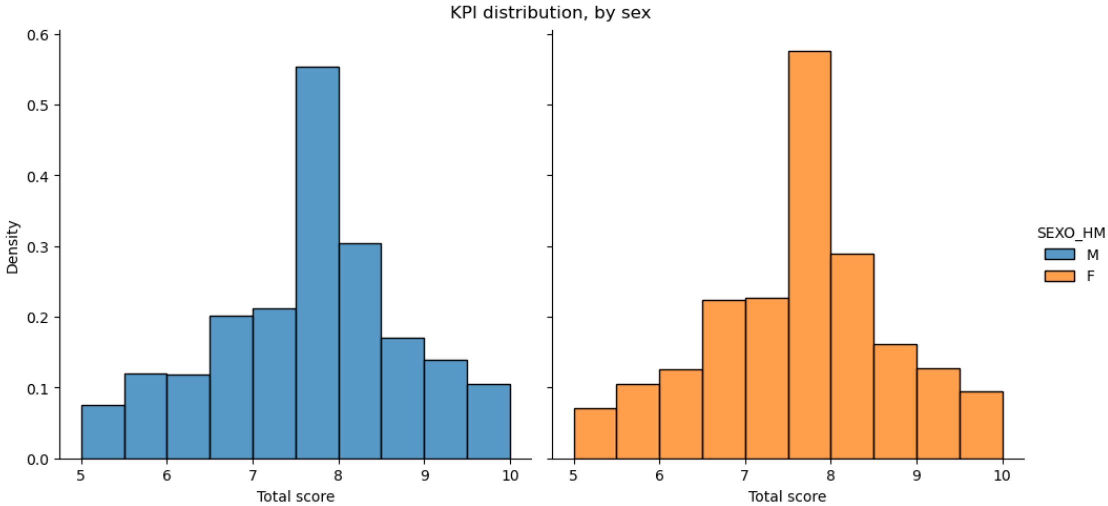
[53]:
```
Text(0.5, 1.0, 'Debt likelihood: boxplot, by education')
```



1.9.4 Filtered histograms

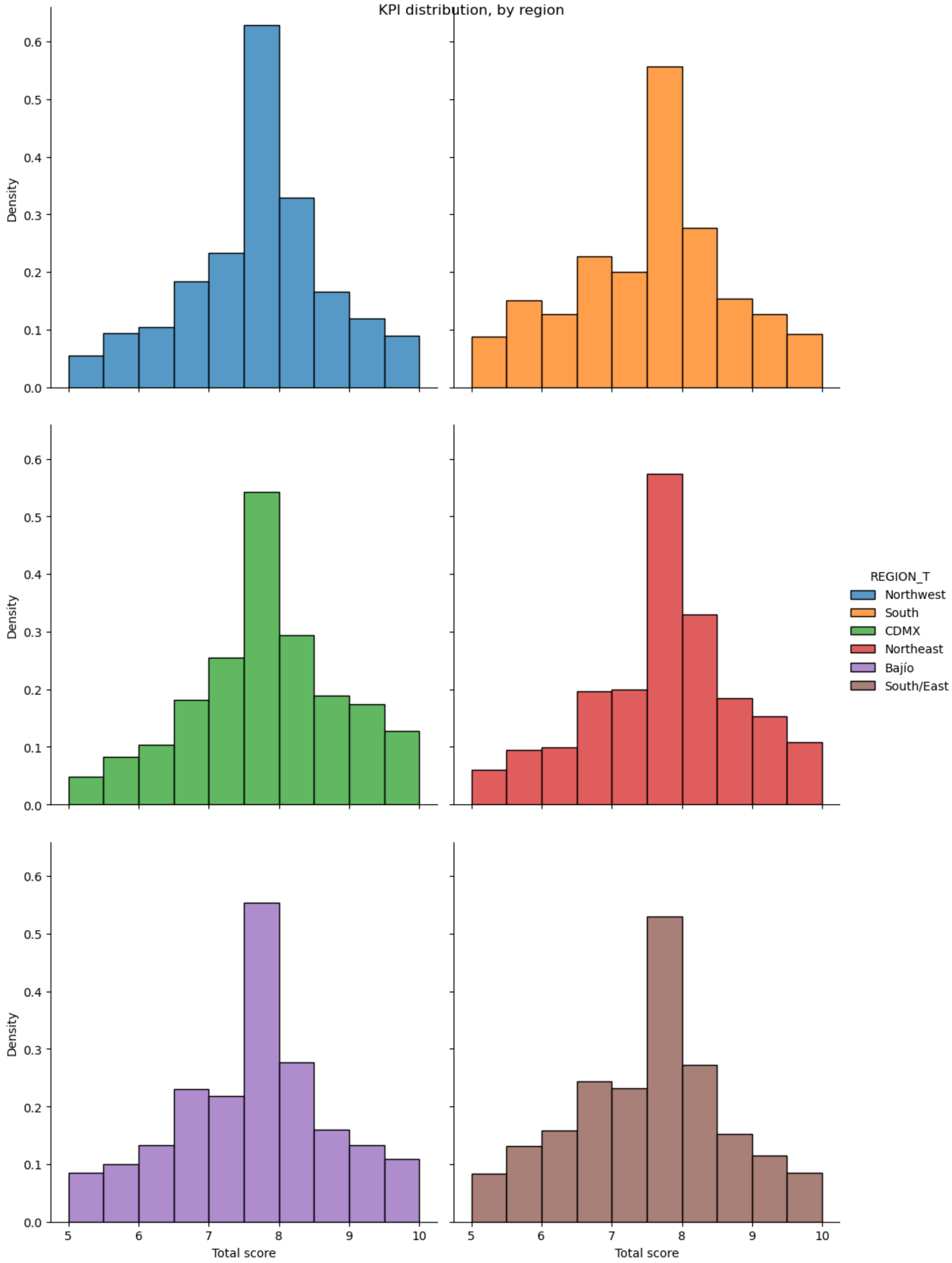
[54]:
```
### Filtered histograms

### By sex

g = sns.FacetGrid(data3, col="SEXO_HM", hue="SEXO_HM", height = 5)
g.map(sns.histplot, f't4{numss}_calif', stat='density', bins=10)
g.set_axis_labels("Total score", "Density")
g.set_titles('')
g.fig.suptitle('KPI distribution, by sex')
g.add_legend()
```

[54]: <seaborn.axisgrid.FacetGrid at 0x2018f59dd30>



[56]:
```
### Filtered histograms

### By region

g = sns.FacetGrid(data3, col="REGION_T", hue="REGION_T", height = 5, col_wrap =␣
↪2)
g.map(sns.histplot, f't4{numss}_calif', stat='density', bins=10)
g.set_axis_labels("Total score", "Density")
g.set_titles('')
g.fig.suptitle('KPI distribution, by region')
g.add_legend()
```

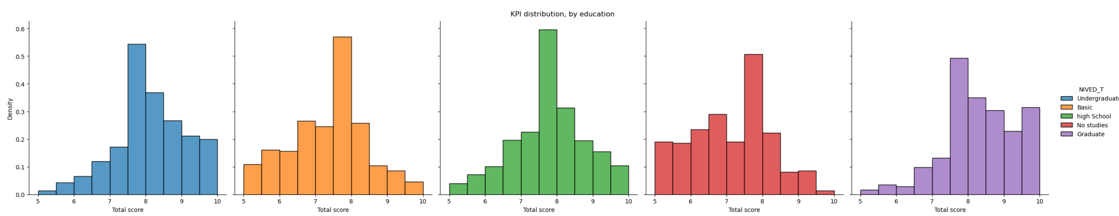
[56]: <seaborn.axisgrid.FacetGrid at 0x2018fcacf20>



[57]:
```
### Filtered histograms

### By region
```

```
g = sns.FacetGrid(data3, col="NIVED_T", hue="NIVED_T", height = 5)
g.map(sns.histplot, f't4{numss}_calif', stat='density', bins=10)
g.set_axis_labels("Total score", "Density")
g.set_titles('')
g.fig.suptitle('KPI distribution, by education')
g.add_legend()
```

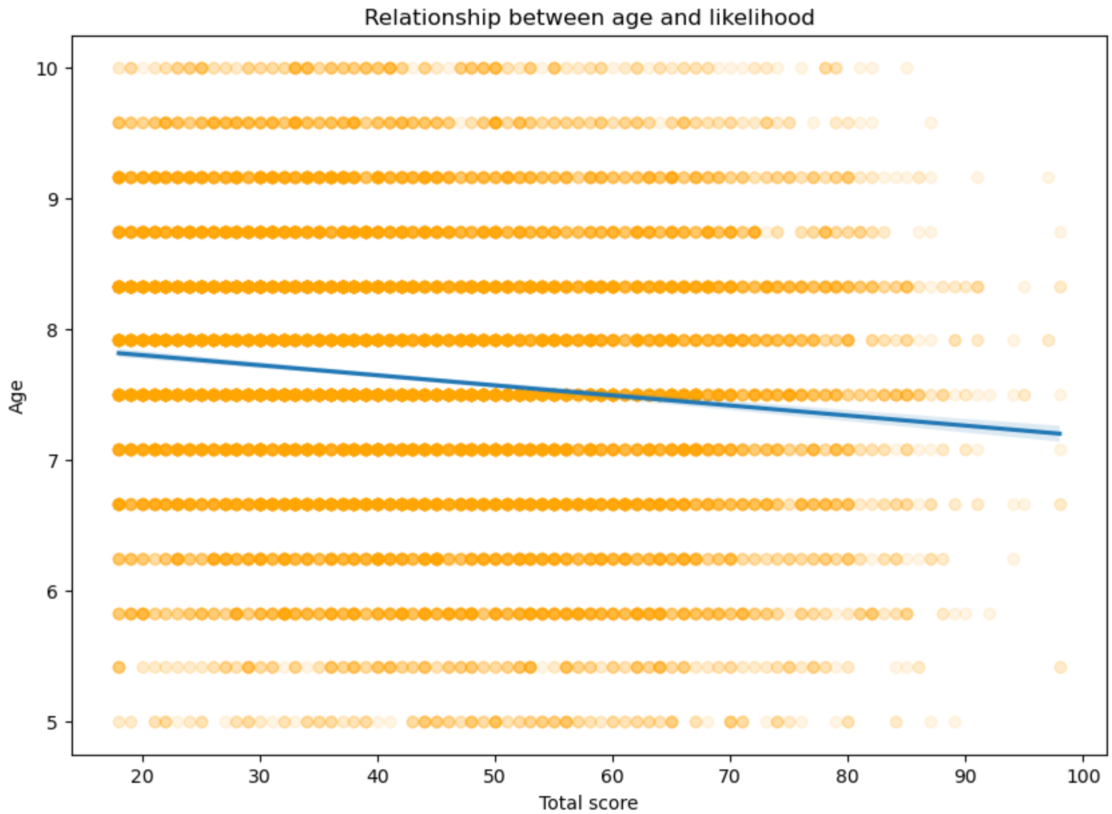
[57]: <seaborn.axisgrid.FacetGrid at 0x2018a266240>



1.9.5 Age and debt likelihood

[58]:
```
fig, ax = plt.subplots(figsize=(10,7))

plot1 = sns.regplot(data3, x = 'EDAD_V', y = f't4{numss}_calif',␣
↪scatter_kws=dict(color="orange", alpha = 0.1))

ax.set_xlabel('Total score')
ax.set_ylabel('Age')
ax.set_title('Relationship between age and likelihood')
```

[58]: Text(0.5, 1.0, 'Relationship between age and likelihood')



We can draw some conclusions: - The Mexican population has mixed expectations towards their economic stability, with a KPI median of 7.5 en su KPI, and no outliers. - There aren't any significative differences between men and women regarding the KPI. - There are some differences in KPI distributions amongst education levels: undergraduate and graduate subpopulations tend to have a more positive perspective of their economic future (MED = 8.33). - With respect to regions, there are two subpopulations with higher-than-average expectations: CDMX and the Northeast (MED = 8).

Back to the Intro

1.10 Predictive analysis

Now we try to predict the answers to the questions given the independent variables: sex, age, education, and locality. A posteriori, we may exclude one or more variables if they are not seen to be significative in predictive power.

1.10.1 Classification model

We will determine which answers, and with which certainty, can be predicted using the independent variables. To do this, we consider three different ML methods: logistic regression, decision trees, and K nearest neighbours.

Logistic regression

[67]:

```
### No predictive power

### Question

numq = 6

### Independent variables

X = data3[['SEXO','EDAD_V', 'NIV', 'TLOC']].to_numpy()

### Dependient variable

Y = data3[f'p4{numss}{numq}_t']

### Training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X,Y,test_size=0.3,␣
↪random_state=42)

### Method

pipeline = Pipeline(steps=[("scaler", StandardScaler()), ("lr",␣
↪LogisticRegression(max_iter=10000000))])
pipeline.fit(X_train,y_train)

### Confusion matrix

y_pred = pipeline.predict(X_test)
print(confusion_matrix(y_test, y_pred))
```

```
[[986 483  12]
 [732 580   7]
 [777 353   5]]
```

[68]:

```
### classification report

print(classification_report(y_test,y_pred))
```

```
               precision    recall  f1-score   support

   De acuerdo       0.40      0.67      0.50      1481
En desacuerdo       0.41      0.44      0.42      1319
      Ninguna       0.21      0.00      0.01      1135

     accuracy                           0.40      3935
    macro avg       0.34      0.37      0.31      3935
 weighted avg       0.35      0.40      0.33      3935
```

Decision tree

[69]:
```
### No predictive power

### Question

numq = 6

### Independent variables

X = data3[['SEXO','EDAD_V', 'NIV', 'TLOC']].to_numpy()

### Dependient variable

Y = data3[f'p4{numss}{numq}_t']

### Training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X,Y,test_size=0.3,␣
    ↪random_state=42)

### Method

pipeline = Pipeline(steps=[("scaler", StandardScaler()), ("tree",␣
    ↪DecisionTreeClassifier())])
pipeline.fit(X_train, y_train)

y_pred = pipeline.predict(X_test)
print(confusion_matrix(y_test, y_pred))
```

```
[[811 439 231]
 [657 475 187]
 [590 375 170]]
```

[70]:
```
print(classification_report(y_test,y_pred))
```

```
               precision    recall  f1-score   support

  De acuerdo        0.39      0.55      0.46      1481
En desacuerdo       0.37      0.36      0.36      1319
     Ninguna        0.29      0.15      0.20      1135

    accuracy                            0.37      3935
   macro avg        0.35      0.35      0.34      3935
weighted avg        0.36      0.37      0.35      3935
```

K Nearest Neighbours

[75]:
```
### No predictive power

### Question

numq = 6

### Independent variables

X = data3[['SEXO','EDAD_V', 'NIV', 'TLOC']].to_numpy()

### Dependient variable

Y = data3[f'p4{numss}{numq}_t']

### Training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X,Y,test_size=0.3,␣
↪random_state=42)

### Method

pipeline = Pipeline(steps=[("scaler", StandardScaler()), ("knn",␣
↪KNeighborsClassifier(n_neighbors=20))])
pipeline.fit(X_train,y_train)

y_pred = pipeline.predict(X_test)
print(confusion_matrix(y_test, y_pred))
```

```
[[829 503 149]
 [586 611 122]
 [566 439 130]]
```

[76]:
```
print(classification_report(y_test,y_pred,zero_division = 0.0))
```

```
               precision    recall  f1-score   support

   De acuerdo       0.42      0.56      0.48      1481
En desacuerdo       0.39      0.46      0.43      1319
      Ninguna       0.32      0.11      0.17      1135

     accuracy                           0.40      3935
    macro avg       0.38      0.38      0.36      3935
 weighted avg       0.38      0.40      0.37      3935
```

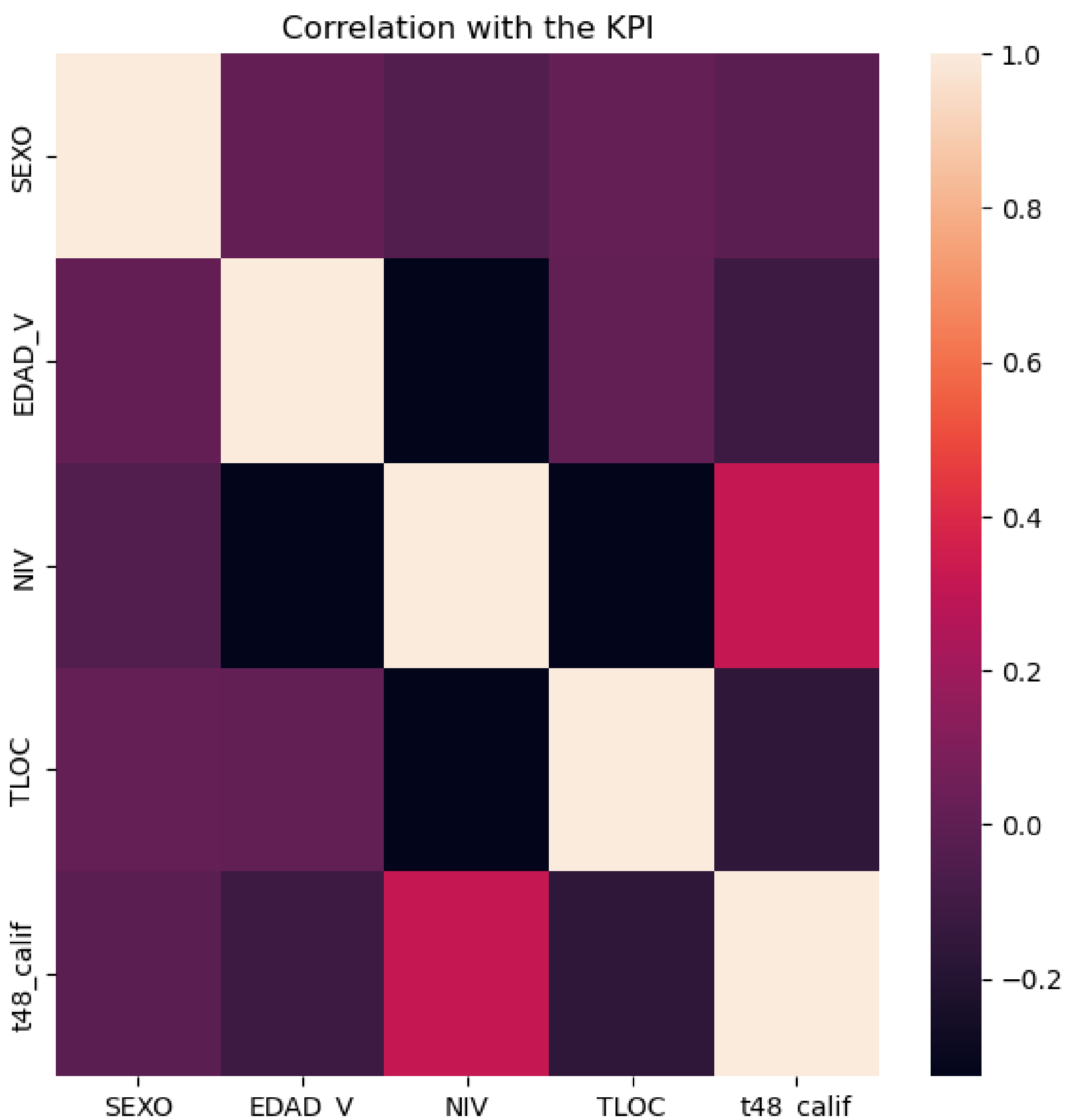
Our analysis in inconclusive as accuracy scores are all less than 50%. This is possibly due to the semi-categorical nature of the independent variables and/or unbalanced data, i.e. overrepresented answers. More advanced techniques are required.

1.10.2 Regression model

Now we wish to predict debt likelihood using the independent variables. First, we wish to determine any obvious variable with a significative contribution on the target variable.

[60]:
```
corr2 = data3[['SEXO', 'EDAD_V', 'NIV', 'TLOC', f't4{numss}_calif']].corr()

fig, ax = plt.subplots(figsize=(7,7))

# plot the heatmap
sns.heatmap(corr2)

ax.set_title('Correlation with the KPI')
```

[60]:
```
Text(0.5, 1.0, 'Correlation with the KPI')
```



[61]:
```
corr2.head(5)
```

[61]:
```
                SEXO    EDAD_V       NIV      TLOC  t48_calif
SEXO       1.000000  0.014057 -0.041977  0.019067  -0.014119
EDAD_V     0.014057  1.000000 -0.327001  0.007592  -0.118083
NIV       -0.041977 -0.327001  1.000000 -0.324332   0.320400
TLOC       0.019067  0.007592 -0.324332  1.000000  -0.163955
t48_calif -0.014119 -0.118083  0.320400 -0.163955   1.000000
```

We find a weak positive correlation between education and the KPI, as previously suggested.

[62]:
```
X = data3[['EDAD_V', 'NIV','TLOC']].to_numpy()

Y = data3[f't4{numss}_calif']

X_train, X_test, y_train, y_test = train_test_split(X,Y,test_size=0.3,␣
 ↪random_state=42)

### Model

lr = LinearRegression()
lr.fit(X_train, y_train)

y_pred = lr.predict(X_test)
```

[63]:
```
### Coefficients

print(lr.coef_)

### r2 score

print(lr.score(X_test, y_test))
```

```
[-0.00179263  0.12366933 -0.05964403]
0.11113827773501239
```

[64]:
```
### Obtaining the errors

mae = mean_absolute_error(y_true=y_test,y_pred=y_pred)
mse = mean_squared_error(y_true=y_test,y_pred=y_pred)
rmse = root_mean_squared_error(y_true=y_test,y_pred=y_pred)

print(f'Mean absolute error: {mae}')
print(f'Mean squared error: {mse}')
print(f'Root mean squared error: {rmse}')
```

```
Mean absolute error: 0.8377961638942412
Mean squared error: 1.0643339744596743
Root mean squared error: 1.0316656311323327
```

Although we claimed that education was the most significative variable regarding debt likelihood, we conclude that the total score cannot be accurately predicted using the independent variables. This was expected as our classification task was also inconclusive, and because a previous visualization showed no observable trend between age and the KPI.

Back to the Intro

1.11 Summary

- Overall, the answers to the survey have similar distributions with respect to sex, region, and education.
- The answers have weak correlations with the independent variables.
- The debt likelihood score, defined as an aggregate of the answers to the survey, does show differences amongst regions and education levels.
- Our ML analysis is not conclusive neither for the answers nor for the KPI, possibly due to overrepresentation for some answers.

1.12 Key takeaways

- The answers to the questions of Subsection 4.8 of the ENIF exhibit a high degree of homogeneity with respect to various socioeconomic descriptors, such as sex, age, region, and education.
- By defining and aggregate of the answers, we can observe a higher notion of economic stability from certain subpopulations, such as graduates and undergraduates when compared to other education levels.
- Similarly, people from CDMX and the Northeast report a lower likelihood of contracting debt.
- The distribution of the answers does not allow us to predict with certainty economic behaviours regarding contraction of debt based on sex, age, locality, and education.

Back to the Intro