READING PDF FILES

REBECCA DUONG

- Goals:
  - Extract the tables and the data from Pdf
  - Find the country with the highest % of Child Marriage by age 15 and 18 (using the data from the textbook by Jacqueline Kazil, Katherine Jarmul)
- Python Libraries that are availble for reading PDF files:
  - pdfminer
  - pdftotext
  - tabula-py
  - PyPDF2
  - pdftables
- The library I will use for the tutorial:
  - PyPDF2
  - pdftables
  - matplotlib
  - pandas

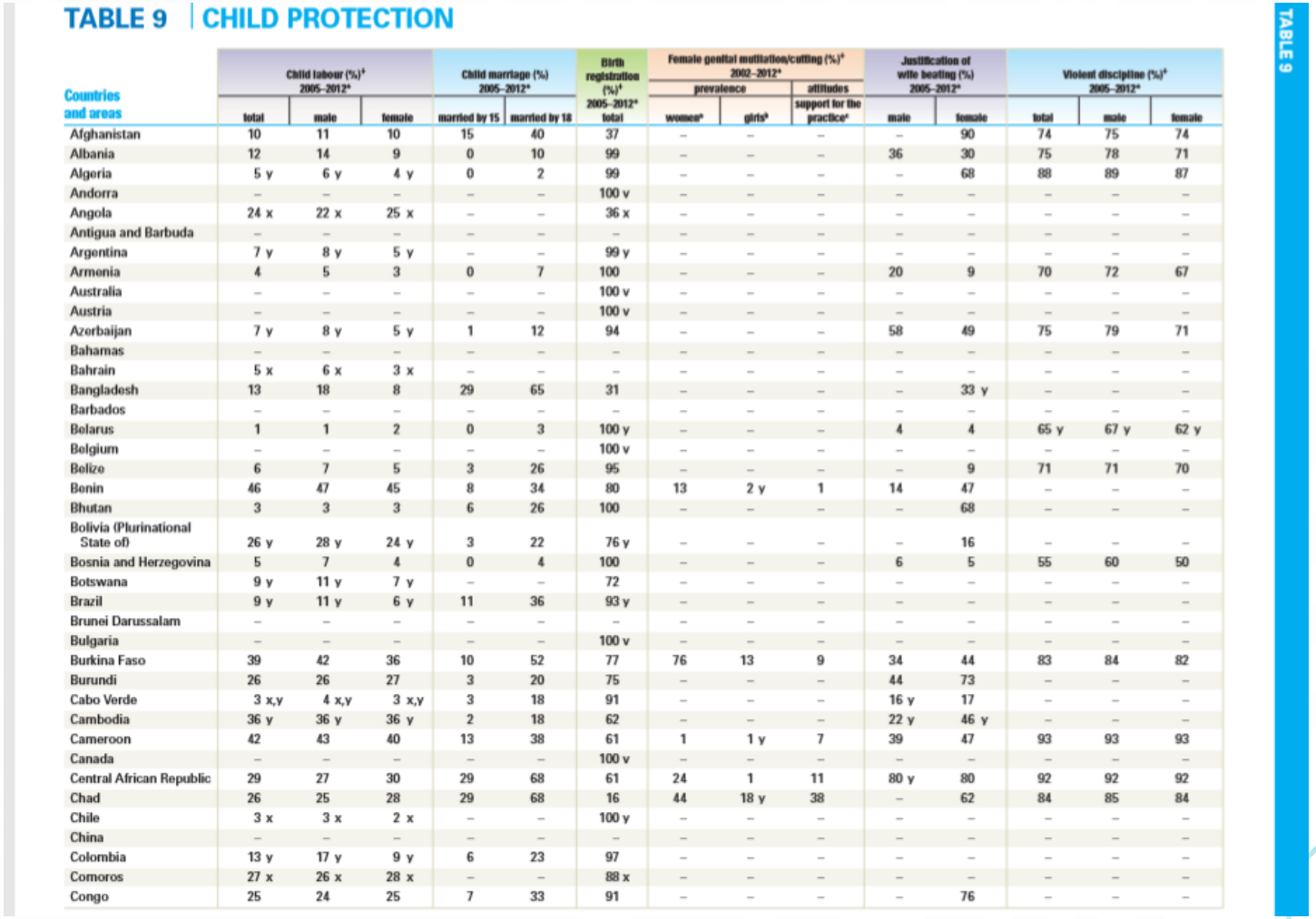


TABLE 9 | CHILD PROTECTION

Countries and areas	Child labour (%)[a] 2005–2012* total	male	female	Child marriage (%) 2005–2012* married by 15	married by 18	Birth registration (%)[b] 2005–2012* total	Female genital mutilation/cutting (%)[c] 2002–2012* prevalence: women[a]	girls[b]	attitudes: support for the practice[c]	Justification of wife beating (%) 2005–2012* male	female	Violent discipline (%)[d] 2005–2012* total	male	female
Afghanistan	10	11	10	15	40	37	–	–	–	–	90	74	75	74
Albania	12	14	9	0	10	99	–	–	–	36	30	75	78	71
Algeria	5 y	6 y	4 y	0	2	99	–	–	–	–	68	88	89	87
Andorra	–	–	–	–	–	100 v	–	–	–	–	–	–	–	–
Angola	24 x	22 x	25 x	–	–	36 x	–	–	–	–	–	–	–	–
Antigua and Barbuda	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Argentina	7 y	8 y	5 y	–	–	99 y	–	–	–	–	–	–	–	–
Armenia	4	5	3	0	7	100	–	–	–	20	9	70	72	67
Australia	–	–	–	–	–	100 v	–	–	–	–	–	–	–	–
Austria	–	–	–	–	–	100 v	–	–	–	–	–	–	–	–
Azerbaijan	7 y	8 y	5 y	1	12	94	–	–	–	58	49	75	79	71
Bahamas	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Bahrain	5 x	6 x	3 x	–	–	–	–	–	–	–	–	–	–	–
Bangladesh	13	18	8	29	65	31	–	–	–	–	33 y	–	–	–
Barbados	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Belarus	1	1	2	0	3	100 y	–	–	–	4	4	65 y	67 y	62 y
Belgium	–	–	–	–	–	100 v	–	–	–	–	–	–	–	–
Belize	6	7	5	3	26	95	–	–	–	–	9	71	71	70
Benin	46	47	45	8	34	80	13	2 y	1	14	47	–	–	–
Bhutan	3	3	3	6	26	100	–	–	–	–	68	–	–	–
Bolivia (Plurinational State of)	26 y	28 y	24 y	3	22	76 y	–	–	–	–	16	–	–	–
Bosnia and Herzegovina	5	7	4	0	4	100	–	–	–	6	5	55	60	50
Botswana	9 y	11 y	7 y	–	–	72	–	–	–	–	–	–	–	–
Brazil	9 y	11 y	6 y	11	36	93 y	–	–	–	–	–	–	–	–
Brunei Darussalam	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Bulgaria	–	–	–	–	–	100 v	–	–	–	–	–	–	–	–
Burkina Faso	39	42	36	10	52	77	76	13	9	34	44	83	84	82
Burundi	26	26	27	3	20	75	–	–	–	44	73	–	–	–
Cabo Verde	3 x,y	4 x,y	3 x,y	3	18	91	–	–	–	16 y	17	–	–	–
Cambodia	36 y	36 y	36 y	2	18	62	–	–	–	22 y	46 y	–	–	–
Cameroon	42	43	40	13	38	61	1	1 y	7	39	47	93	93	93
Canada	–	–	–	–	–	100 v	–	–	–	–	–	–	–	–
Central African Republic	29	27	30	29	68	61	24	1	11	80 y	80	92	92	92
Chad	26	25	28	29	68	16	44	18 y	38	–	62	84	85	84
Chile	3 x	3 x	2 x	–	–	100 y	–	–	–	–	–	–	–	–
China	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Colombia	13 y	17 y	9 y	6	23	97	–	–	–	–	–	–	–	–
Comoros	27 x	26 x	28 x	–	–	88 x	–	–	–	–	–	–	–	–
Congo	25	24	25	7	33	91	–	–	–	–	76	–	–	–

USING PDFTABLES

```
from pdftables import get_tables
import pprint


# The get_tables function return each pages as its own table
# Each of those tables has a list ofrows with a contained list of columns
all_tables = get_tables(open('EN-FINAL Table 9.pdf', 'rb'))


# Tying to find the titles we can use for our columns
# Here we are looking at the first page's first 6 rows
print all_tables[0][:6]

# NOTE: we can see that the titles are included in the first 3 list and they are messy


# Manually settting up our titles
# So we add all of our headers, including the country names, to one list
headers = ['Country', 'Child Labor 2005-2012 (%) total',
           'Child Labor 2005-2012 (%) male',
           'Child Labor 2005-2012 (%) female',
           'Child Marriage 2005-2012 (%) married by 15',
           'Child Marriage 2005-2012 (%) married by 18',
           'Birth registration 2005-2012 (%)',
           'Female Genital mutilation 2002-2012 (prevalence), women',
           'Female Genital mutilation 2002-2012 (prevalence), girls',
           'Female Genital mutilation 2002-2012 (support)',
           'Justification of wife beating 2005-2012 (%) male',
           'Justification of wife beating 2005-2012 (%) female',
           'Violent discipline 2005-2012 (%) total',
           'Violent discipline 2005-2012 (%) male',
           'Violent discipline 2005-2012 (%) female']
```

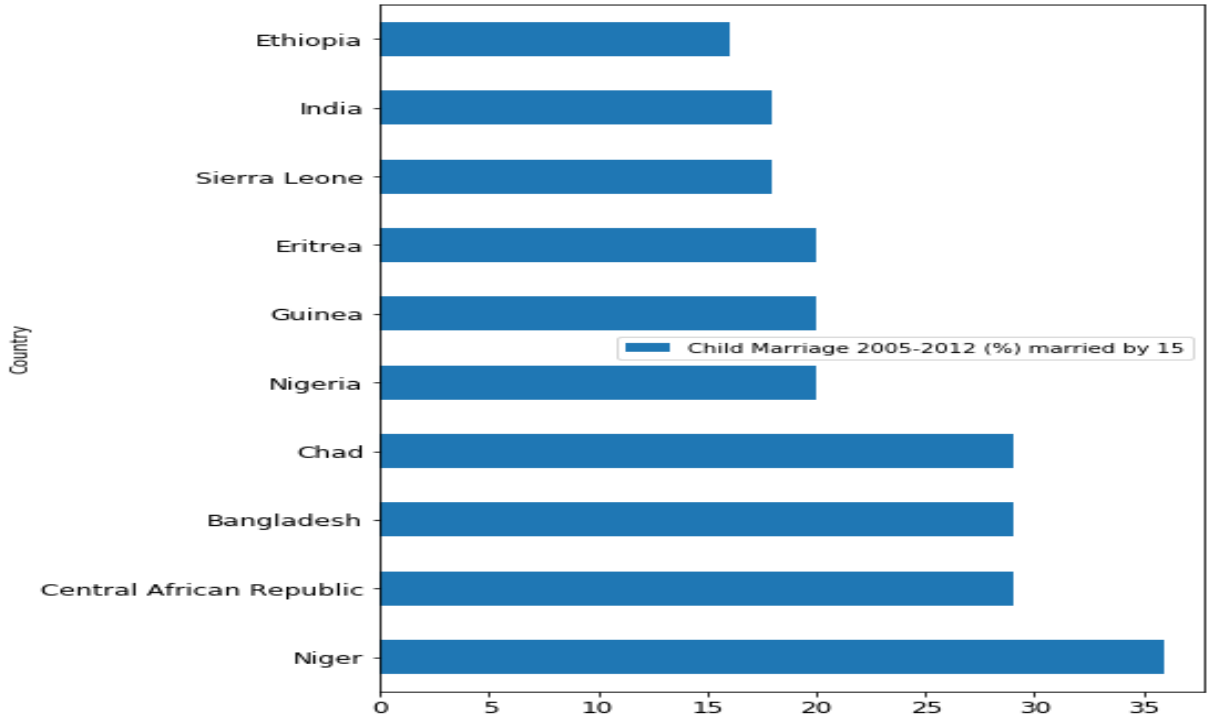
```
# extract the data values from the pdf
first_name = False
final_data = []

for table in all_tables:

    # isolates only the rows for each page we want; slice from the 5th index onward
    for row in table[5:]:

        # if data row is missing index 0, it has no country name and is a blank row
        # skip this row using continue
        if row[0] == '' or row[0][0].isdigit():
            continue

        # if the data row is missing index 2, we know this is probably the first part of a country name
        # We saves the first part of the name in a variable first_name
        elif row[2] == '':
            first_name = row[0]
            continue

        # Manipulate the country name entry in the row if it has a first_name
        # Put the second part of a country name back together with the first name
        if first_name:
            row[0] = u'{} {}'.format(first_name, row[0])

            # set first_name back to False, so our nexxt iteration operates properly
            first_name = False

        # Before cleaning up the messy data...
        # Note there are in unicode
        # print(row)
```

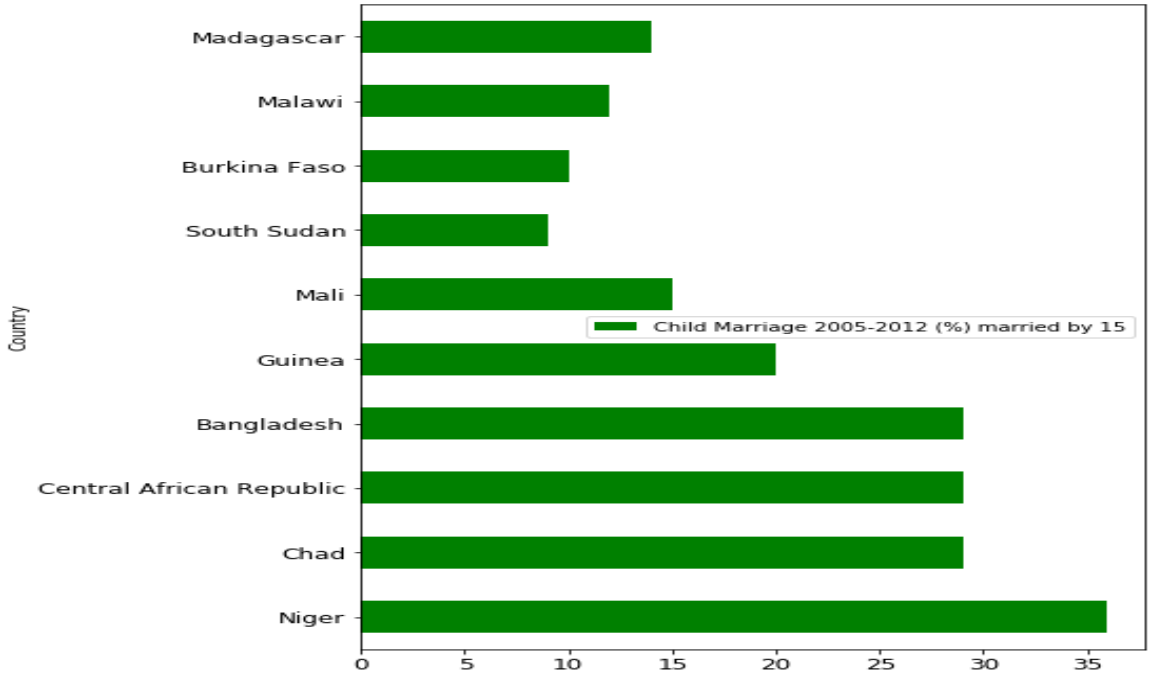
```
[u'Afghanistan', u'10', u'11', u'10', u'15', u'40', u'37', u'\u2013', u'\u2013', u'\u2013', u'90', u'74', u'75', u
[u'Albania', u'12', u'14', u'9', u'0', u'10', u'99', u'\u2013', u'\u2013', u'\u2013', u'36', u'30', u'75', u'78', u'71']
[u'Algeria', u'5 y', u'6 y', u'4 y', u'0', u'2', u'99', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'68', u'88', u'89', u'87']
[u'Andorra', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'100 v', u'\u2013', u'\u2013', u'\u2013', u'\u201
[u'Angola', u'24 x', u'22 x', u'25 x', u'\u2013', u'\u2013', u'36 x', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013
[u'Antigua and Barbuda', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u
[u'Argentina', u'7 y', u'8 y', u'5 y', u'\u2013', u'\u2013', u'99 y', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013
[u'Armenia', u'4', u'5', u'3', u'0', u'7', u'100', u'\u2013', u'\u2013', u'\u2013', u'20', u'9', u'70', u'72', u'67']
[u'Australia', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'100 v', u'\u2013', u'\u2013', u'\u2013', u'\u201
[u'Austria', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'100 v', u'\u2013', u'\u2013', u'\u2013', u'\u2013
[u'Azerbaijan', u'7 y', u'8 y', u'5 y', u'1', u'12', u'94', u'\u2013', u'\u2013', u'\u2013', u'58', u'49', u'75', u'79', u'71']
[u'Bahamas', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2
[u'Bahrain', u'5 x', u'6 x', u'3 x', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013
[u'Bangladesh', u'13', u'18', u'8', u'29', u'65', u'31', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'33 y', u'\u2013', u'
[u'Barbados', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2
[u'Belarus', u'1', u'1', u'2', u'0', u'3', u'100 y', u'\u2013', u'\u2013', u'\u2013', u'4', u'4', u'65 y', u'67 y', u'62 y']
[u'Belgium', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'100 v', u'\u2013', u'\u2013', u'\u2013', u'\u201
[u'Belize', u'6', u'7', u'5', u'3', u'26', u'95', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'9', u'71', u'71', u'70']
[u'Benin', u'46', u'47', u'45', u'8', u'34', u'80', u'13', u'2 y', u'1', u'14', u'47', u'\u2013', u'\u2013', u'\u2013']
[u'Bhutan', u'3', u'3', u'3', u'6', u'26', u'100', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'68', u'\u2013', u'\u2013',
[u'Bolivia (Plurinational Bolivia (Plurinational State of)', u'26 y', u'28 y', u'24 y', u'3', u'22', u'76 y', u'\u2013', u'\u2
[u'Bosnia and Herzegovina', u'5', u'7', u'4', u'0', u'4', u'100', u'\u2013', u'\u2013', u'\u2013', u'6', u'5', u'55', u'60',
[u'Botswana', u'9 y', u'11 y', u'7 y', u'\u2013', u'\u2013', u'72', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013']
[u'Brazil', u'9 y', u'11 y', u'6 y', u'11', u'36', u'93 y', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013']
[u'Brunei Darussalam', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013']
[u'Bulgaria', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'100 v', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013']
[u'Burkina Faso', u'39', u'42', u'36', u'10', u'52', u'77', u'76', u'13', u'9', u'34', u'44', u'83', u'84', u'82']
[u'Burundi', u'26', u'26', u'27', u'3', u'20', u'75', u'\u2013', u'\u2013', u'\u2013', u'44', u'73', u'\u2013', u'\u2013', u'\u2013']
[u'Cabo Verde', u'3 x,y', u'4 x,y', u'3 x,y', u'3', u'18', u'91', u'\u2013', u'\u2013', u'\u2013', u'16 y', u'17', u'\u2013', u'\u2013', u'\u2013']
[u'Cambodia', u'36 y', u'36 y', u'36 y', u'2', u'18', u'62', u'\u2013', u'\u2013', u'\u2013', u'22 y', u'46 y', u'\u2013', u'\u2013', u'\u2013']
[u'Cameroon', u'42', u'43', u'40', u'13', u'38', u'61', u'1', u'1 y', u'7', u'39', u'47', u'93', u'93', u'93']
[u'Canada', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'100 v', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013', u'\u2013']
```

```
[['Afghanistan', 10, 11, 10, 15, 40, 37, '', '', '', '', 90, 74, 75, 7
 ['Albania', 12, 14, 9, 0, 10, 99, '', '', '', 36, 30, 75, 78, 71],
 ['Algeria', '5 ', '6 ', '4 ', 0, 2, 99, '', '', '', '', 68, 88, 89, 87],
 ['Andorra', '', '', '', '', '', '100 ', '', '', '', '', '', '', '', ''],
 ['Angola',
  '24 ',
  '22 ',
  '25 ',
  '',
  '',
  '36 ',
  '',
  '',
  '',
  '',
  '',
  '',
  '',
```

```
    clean_row = []
    # clean the data value in each row
    # strip away the non-numeric letter next to the numeric data in each column such as 'x', 'y', 'v', or x,y'
    # also change the unicode representation of '-' into an empty string
    for col in range(0, len(row)):

      # Translate the value from unicode to ascii and do not need to transalte the non-ASCII chars
      # store the data values into the variable clean_row
      # https://stackoverflow.com/questions/2365411/convert-unicode-to-ascii-without-errors-in-python/35536228
      clean_row.append(row[col].encode('ascii', 'ignore'))

      # Then clean up the data values
      # strip away the non-numeric letter next to the numeric value
      clean_row[col] = clean_row[col].strip('x')
      clean_row[col] = clean_row[col].strip('y')
      clean_row[col] = clean_row[col].strip('v')
      clean_row[col] = clean_row[col].strip('x,y')

      # convert the numeric data for each row to integer
      if clean_row[col].isdigit():
          clean_row[col] = eval(clean_row[col])
          # print(type(clean_row[col]))
      elif clean_row[col] == ' ':
          clean_row[col] = None

    # print(clean_row)

    # append all the cleaned row into the list final_data
    final_data.append(clean_row)

    # Exit the loop when you reach the last data row
    if row[0] == 'Zimbabwe':
        break
```

```
# prints out a list of each row in the table
pprint.pprint(final_data)
```






```
import pandas as pd
```

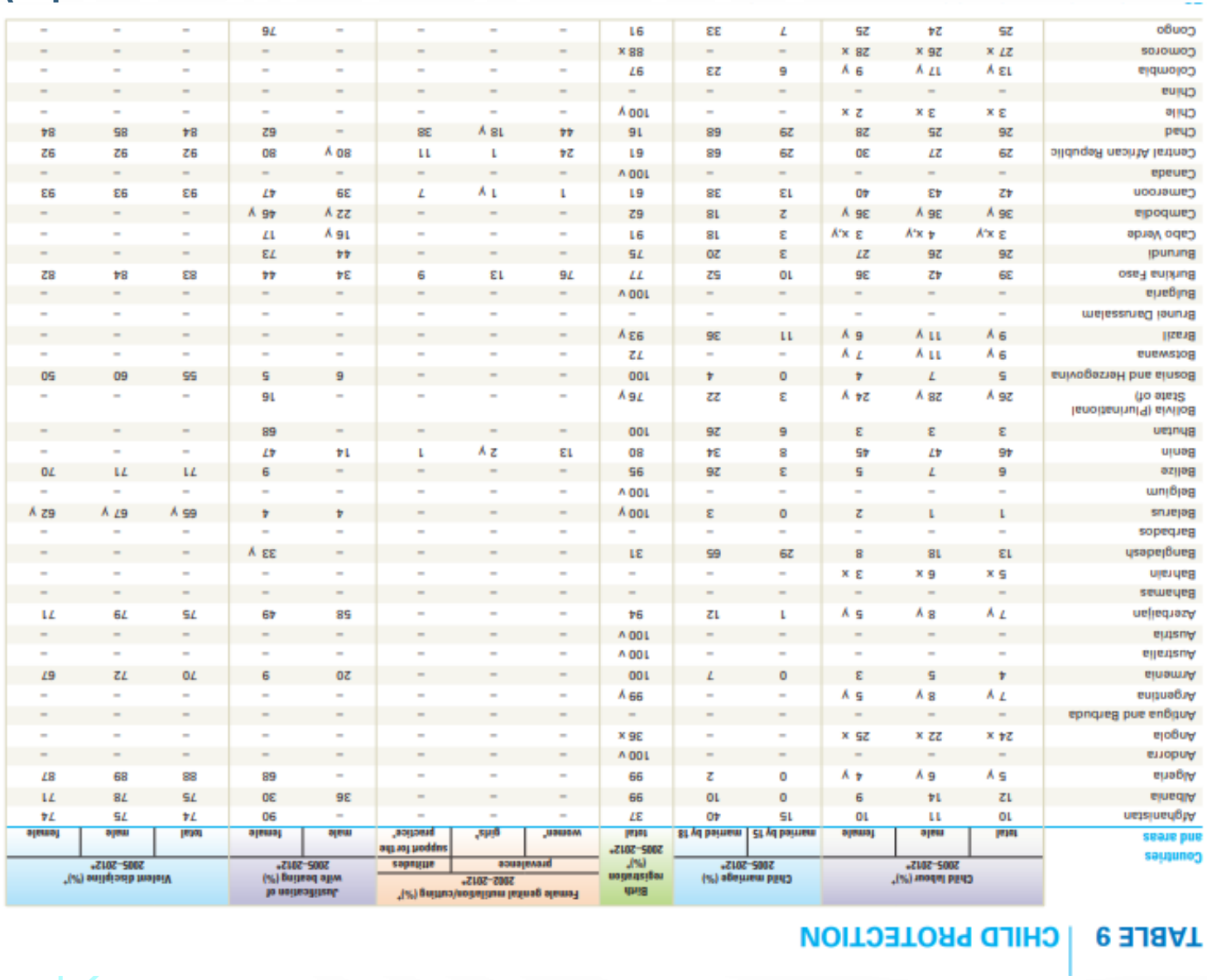
```
# convert the list of dict into a dataframe and set the 'Country' column as the index
df = pd.DataFrame(final_data, columns = headers)
df = df.set_index('Country')
df = df.apply(pd.to_numeric)
# df.columns
df
```

```
import matplotlib.pyplot as plt

# Find the country with the highest % of child marriage in 2005-2012 of age 15 and 18

fig, axes = plt.subplots(nrows = 1, ncols = 2)

# adjust the space between the 2 plots; set it a little more farther apart
plt.subplots_adjust(wspace = 1)
```

```
# Sort the dataframe according to its % of child marriage by age 15 and 18; the highest % will be at th
# store these into a new variable
df_1 = df.sort_values('Child Marriage 2005-2012 (%) married by 15', ascending = False)
# df_1
df_2 = df.sort_values('Child Marriage 2005-2012 (%) married by 18', ascending = False)
# df_2

# Looking at the data, it seems like Niger the highest % of Child Marriage in 2005-2012 for both age 1
df_1.head(10).plot(ax = axes[0], kind = 'barh', y = ['Child Marriage 2005-2012 (%) married by 15'], font
df_2.head(10).plot(color = 'g', ax = axes[1], kind = 'barh', y = ['Child Marriage 2005-2012 (%) married b
```

```
# Convert the dataframe to csv
df.to_csv('table_9.csv')
```

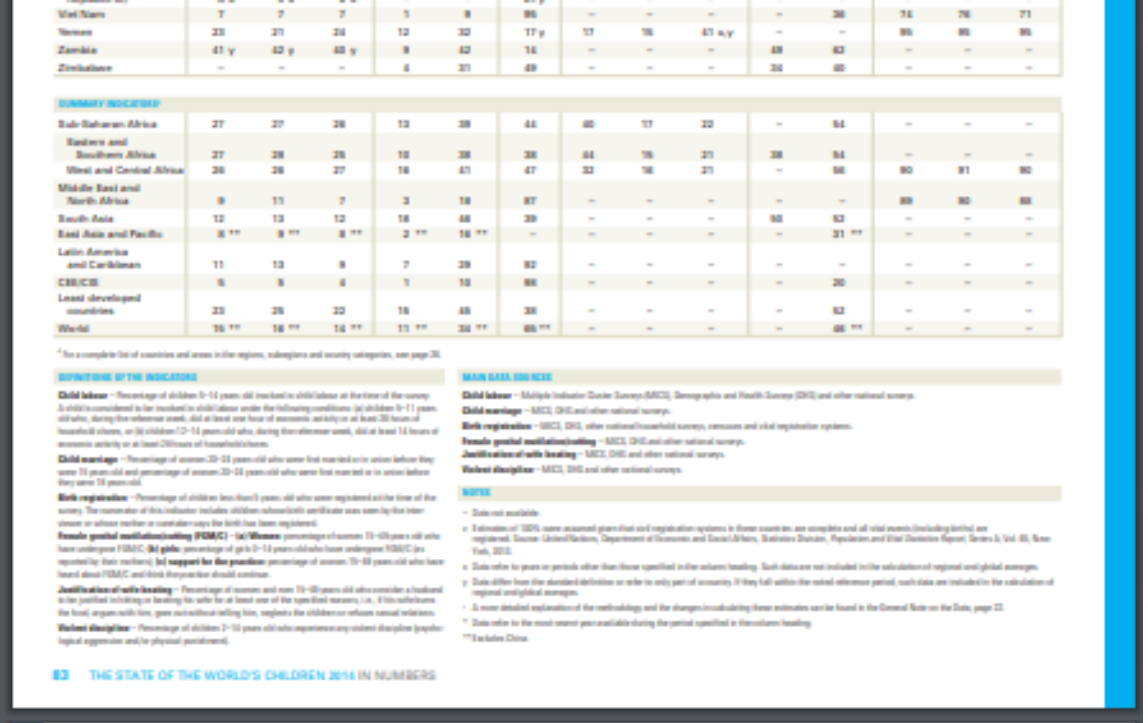
USING PYPDF2 LIBRARY

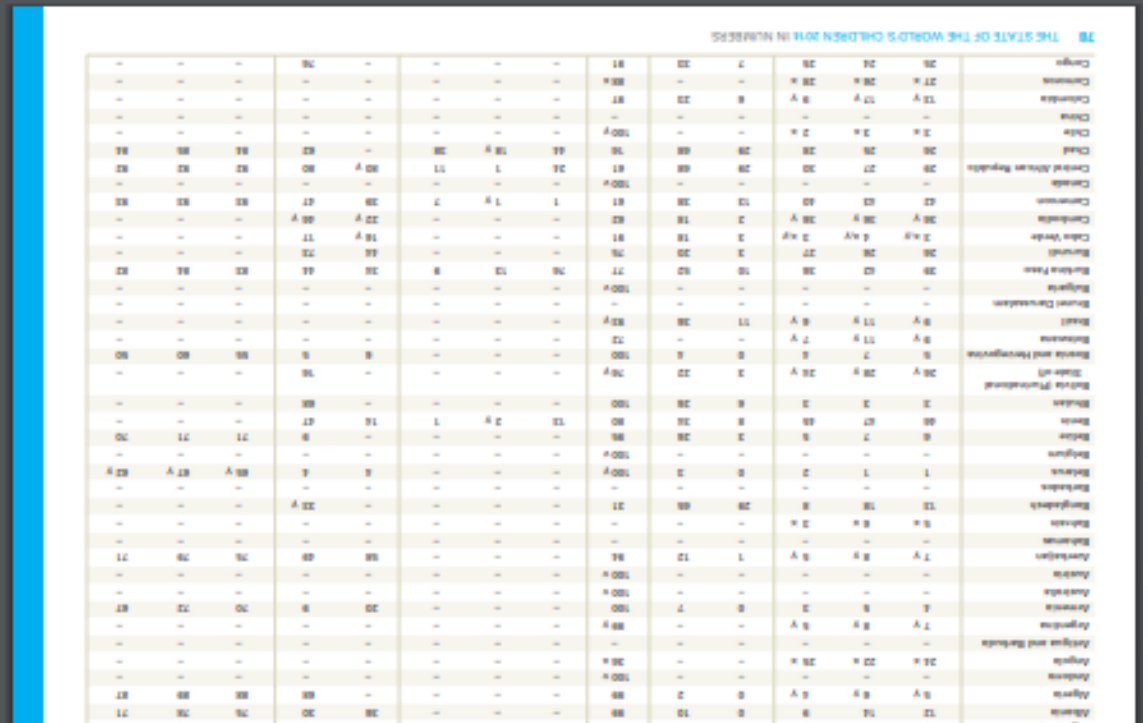


Countries and areas	Child labour (%)* 2005–2012* total	male	female	Child marriage (%) 2005–2012* married by 15	married by 18	Birth registration (%)* 2005–2012* total	Female genital mutilation/cutting (%)* 2002–2012* prevalence women*	girls*	attitudes support for the practice*	Justification of wife beating (%) 2005–2012* male	female	Violent discipline (%)* 2005–2012* total	male	female
Afghanistan	10	11	10	15	40	37	–	–	–	–	90	74	75	74
Albania	12	14	9	0	10	99	–	–	–	36	30	75	78	71
Algeria	5 y	6 y	4 y	0	2	99	–	–	–	–	68	88	89	87
Andorra	–	–	–	–	–	100 v	–	–	–	–	–	–	–	–
Angola	24 x	22 x	25 x	–	–	36 x	–	–	–	–	–	–	–	–
Antigua and Barbuda	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Argentina	7 y	8 y	5 y	–	–	99 y	–	–	–	–	–	–	–	–
Armenia	4	5	3	0	7	100	–	–	–	20	9	70	72	67
Australia	–	–	–	–	–	100 v	–	–	–	–	–	–	–	–
Austria	–	–	–	–	–	100 v	–	–	–	–	–	–	–	–
Azerbaijan	7 y	8 y	5 y	1	12	94	–	–	–	58	49	75	79	71
Bahamas	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Bahrain	5 x	6 x	3 x	–	–	–	–	–	–	–	–	–	–	–
Bangladesh	13	18	8	29	65	31	–	–	–	–	33 y	–	–	–
Barbados	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Belarus	1	1	2	0	3	100 y	–	–	–	4	4	65 y	67 y	62 y
Belgium	–	–	–	–	–	100 v	–	–	–	–	–	–	–	–
Belize	6	7	5	3	26	95	–	–	–	–	9	71	71	70
Benin	46	47	45	8	34	80	13	2 y	1	14	47	–	–	–
Bhutan	3	3	3	6	26	100	–	–	–	–	68	–	–	–
Bolivia (Plurinational State of)	26 y	28 y	24 y	3	22	76 y	–	–	–	–	16	–	–	–
Bosnia and Herzegovina	5	7	4	0	4	100	–	–	–	6	5	55	60	50
Botswana	9 y	11 y	7 y	–	–	72	–	–	–	–	–	–	–	–
Brazil	9 y	11 y	6 y	11	36	93 y	–	–	–	–	–	–	–	–
Brunei Darussalam	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Bulgaria	–	–	–	–	–	100 v	–	–	–	–	–	–	–	–
Burkina Faso	39	42	36	10	52	77	76	13	9	34	44	83	84	82
Burundi	26	26	27	3	20	75	–	–	–	44	73	–	–	–
Cabo Verde	3 x,y	4 x,y	3 x,y	3	18	91	–	–	–	16 y	17	–	–	–
Cambodia	36 y	36 y	36 y	2	18	62	–	–	–	22 y	46 y	–	–	–
Cameroon	42	43	40	13	38	61	1	1 y	7	39	47	93	93	93
Canada	–	–	–	–	–	100 v	–	–	–	–	–	–	–	–
Central African Republic	29	27	30	29	68	61	24	1	11	80 y	80	92	92	92
Chad	26	25	28	29	68	16	44	18 y	38	–	62	84	85	84
Chile	3 x	3 x	2 x	–	–	100 y	–	–	–	–	–	–	–	–
China	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Colombia	13 y	17 y	9 y	6	23	97	–	–	–	–	–	–	–	–
Comoros	27 x	26 x	28 x	–	–	88 x	–	–	–	–	–	–	–	–
Congo	25	24	25	7	33	91	–	–	–	–	76	–	–	–

TABLE 9 | CHILD PROTECTION

```
#A function for rotating the page
def PDFrotate(origFileName, newFileName, rotation):

    # creating a pdf File object of original pdf
    pdfFileObj = open(origFileName, 'rb')

    # creating a pdf Reader object
    pdfReader = PyPDF2.PdfFileReader(pdfFileObj)

    # creating a pdf writer object for new pdf
    # rotated pages will be written to a new pdf
    pdfWriter = PyPDF2.PdfFileWriter()

    # iteriate through each page of original pdf
    for page in range(pdfReader.numPages):

        # creating rotated page object
        pageObj = pdfReader.getPage(page)

        # now we rotate the page by rotateClockwise() method
        pageObj.rotateClockwise(rotation)

        # adding rotated page object to pdf writer
        pdfWriter.addPage(pageObj)

    # open new pdf file object
    newFile = open(newFileName, 'wb')

    # writing rotated pages to new file using write() method
    pdfWriter.write(newFile)

    # closing the original pdf file object
    pdfFileObj.close()

    # closing the new pdf file object
    newFile.close()
```

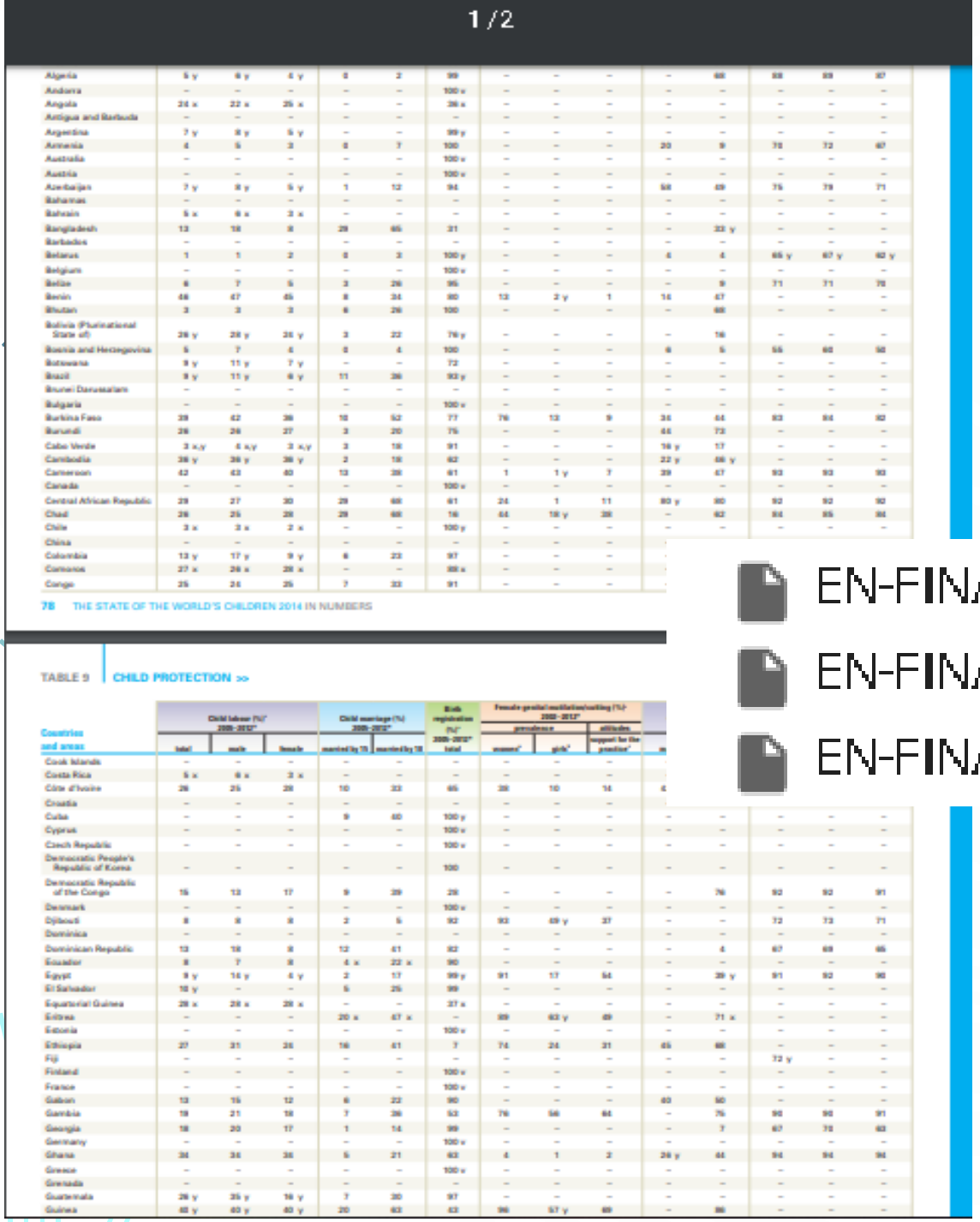




```
# Merging PDF files
```

```
#A function to merge 2 pdfs together

def PDFmerge(pdfs, output):
    # creating pdf file merger object
    pdfMerger = PyPDF2.PdfFileMerger()

    # append file object of each pdf to pdf merger object
    for pdf in pdfs:
        with open(pdf, 'rb') as f:
            pdfMerger.append(f)

    # writing combined pdf to output pdf file
    with open(output, 'wb') as f:
        pdfMerger.write(f)
```

```
# pdf files to merge
pdfs = ['EN-FINAL Table 9.pdf', 'rotated_table_9.pdf']

# output pdf file name
output  = 'combined_table.pdf'

# calling pdf merge function
PDFmerge(pdfs = pdfs, output = output)
```



EN-FINAL Table 90.pdf

EN-FINAL Table 91.pdf

EN-FINAL Table 92.pdf

```
# Splitting PDF file

#A function to split the pdf file

def PDFsplit(pdf, splits):
    # creating input pdf file object
    pdfFileObj = open(pdf, 'rb')

    # creating pdf reader object
    pdfReader = PyPDF2.PdfFileReader(pdfFileObj)

    # starting index of first slice
    start = 0

    # starting index of last slice
    end = splits[0]

    for i in range(len(splits)+1):
                ting pdf writer object for (i+1)th split
                iter = PyPDF2.PdfFileWriter()

                ut pdf file name
                str(i) is the xth number of split we are currently on
                pdf = pdf.split('.pdf')[0] + str(i) + '.pdf'

                ng pages to pdf writer object
                ge in range(start,end):
                Writer.addPage(pdfReader.getPage(page))

        # writing split pdf pages to pdf file
        with open(outputpdf, "wb") as f:
            pdfWriter.write(f)

        # interchanging page split start position for next split
        start = end
        try:
            # setting split end positon for next split
            end = splits[i+1]
        except IndexError:
            # setting split end position for last split
            end = pdfReader.numPages
```

SOURCES/ SHAREABLE LINK

- Shareable Link of my Tutorial:
  - https://colab.research.google.com/drive/1Qm7_N-UzV9E7QucywjfI2A_fpKWYXm3v
- Sources:
  - Jacqueline Kazil, Katherine Jarmul - Data Wrangling with Python (Ch.5)
  - https://www.geeksforgeeks.org/working-with-pdf-files-in-python/
  - https://stackoverflow.com/questions/2365411/convert-unicode-to-ascii-without-errors-in-python/355362