Parental Leave Project Summary

This is a summary of the results for this project. For a more detailed walk through my thought process and the steps involved, see my jupyter notebooks here: https://github.com/anotisberger/Parental-Leave-Project

Introduction

It seems like every day, my newsfeed is flooded with articles about the gender wage gap, the experiences of women in tech fields, and how companies are addressing issues of work-life balance. When my sister gave birth in another country, with very different maternity leave policies than the US, I began thinking about how different government policies could lead to different outcomes for women's engagement in the workforce. In particular, one of my sister's colleagues mentioned that in her home country, maternity leave is a full year, and as a result companies there are reluctant to hire women. This was something I hadn't much considered before, and I thought it might be interesting to examine some data to see if there is a relationship between length of maternity leave and women's employment rates. Of course, there many variables and this is a complicated issue that is difficult to quantify, but I thought this was a good place to start.

Overview

My project consists of two parts:

- Part I – Getting Data: There are several Wikipedia pages with tables of relevant data organized by country, and the Organization for Economic Cooperation and Development (OECD) has an open database of employment data from many countries. I wanted to scrape the data from the Wikipedia pages and put it into .csv files that I could then use and analyze in python.
- Part II – Cleaning and merging the data into pandas dataframes and generating some plots to answer my questions about the relationship between length of maternity leave and women's engagement in the workforce.

Part I – Getting Data

In this part of the project, I wanted to acquire data about the length of maternity leave in different countries, and employment rates broken down by gender in these countries. I found a database created by the Organization for Economic Cooperation and Development (OECD) with employment data from many countries. They have many different measures of employment, broken down by gender, age, and many other factors, but I decided to use full time equivalent employment for all adults broken down by gender as the most straightforward measure of full engagement in the workforce. It was easy enough to download this information in the form of a .csv file from the OECD website.

The bigger challenge would be getting data about maternity leave in different countries in a format that is python-friendly. I found a Wikipedia page with tables of

information about parental leave broken down by country, and I wanted to find a way to extract these tables and put them into .csv files. Luckily, this ended up being easier than I thought, because I found a blog post by Andy Roche about exactly this problem. See here: https://roche.io/2016/05/scrape-wikipedia-with-python
I started out using Andy's code verbatim, but there was a problem with the Parental Leave Wikipedia page – there are 5 tables on the page, and the code was only scraping 3 of them.

The three tables that got scraped are:

- Africa
- Americas
- Europe and Central Asia

The two that got missed are:

- Asia/Pacific
- United Nations

Upon inspecting the webpage, I realized that the three tables that were correctly identified were sortable, meaning you can click on an arrow icon in the column headings to sort the tables in ascending or descending order. In contrast, the two missing tables are not sortable.

I looked more carefully at the HTML code from those two tables to see how it differs from the sortable tables. This is easy to do in Google Chrome by right clicking on the website and choosing "Inspect."

There are two types of tables. In HTML, they look like this:

- Sortable Tables
  `<table class="wikitable sortable jquery-tablesorter">`
- Non-sortable Tables
  `<table class="wikitable">`

I took a second look at Andy's scrape() function – there is some attempt to grab tables of different types. He uses the following lines of code to pick out the tables of interest:

```
table_classes = {"class": ["sortable", "plainrowheaders"]}
wikitables = soup.findAll("table",table_classes)
```

I modified the code to add the class "wikitable" to the list of table classes to scrape.

```
table_classes = {"class": ["sortable", "plainrowheaders", "wikitable"]}
```

and … Success! The new code now successfully scrapes all 5 tables from the Parental Leave webpage.

Part II – Cleaning and Merging Data, Generating Plots

Part IIa – Cleaning Maternity Leave Data

In this part of the project I wanted to combine all of my data using pandas dataframes and create some plots to investigate if there is any relationship between length of maternity leave and employment rates for women.

To start, I had several different tables from the Parental Leave Wikipedia page, representing different regions of the world. I wanted to combine these into one pandas dataframe with two columns, Country, and Maternity Leave (weeks). I first read these files into individual dataframes, using pandas' .read_csv() function. Then, I used pandas.info() to get some information about the tables. Most of the tables already had a column called "Maternity Leave (weeks)," but one of the tables was a little different. ParentalLeave_2 had separate columns for paid and unpaid maternity leave, and the entries are not purely numeric.

Luckily, there were only 8 entries in the "Unpaid maternity leave" column – all the rest were NaN, so I could deal with those problem cases manually.

The rest of the countries, with information only in the "Paid maternity leave" column were still a little messy to deal with. The entries are in string format, and they are not uniform. The length of time is listed in weeks, days, or years. I wrote the following function to convert the entries in this column into a number of weeks:

```
def clean_paid_maternity_leave(string_input):
    count = -1
    num_weeks = []
    input_list = string_input.split(' ')
    for item in input_list:
        count = count+1
        if (item == 'days') & (num_weeks==[]):
            #I assume a 5 day work week in converting the number
            of days into weeks
            num_weeks = float(input_list[count-1])/5
        elif (item == 'weeks') & (num_weeks==[]):
```

```
            num_weeks = float(input_list[count-1])
        elif (item == 'months') & (num_weeks==[]):
            num_weeks = float(input_list[count-1])*4
    return num_weeks
```

Using pandas .apply() function, I applied my function elementwise to the "Paid maternity leave" column, assigning the output to a new row in the dataframe called "Maternity leave (weeks)."

I used the .head() function to look at the first few rows, and the .info() function to get some information about the whole dataframe, now that I have added the new column:

In [18]:
```
ParentalLeave_2.head(5)
```

Out[18]:

	Country	Paid maternity leave	Paid paternity leave	Unpaid maternity leave	Unpaid paternity leave	Restrictions	Maternity leave (weeks)
0	Afghanistan	90 days 100%	NaN	NaN	NaN	NaN	18.0
1	Azerbaijan	126 days 100%	NaN	NaN	NaN	NaN	25.2
2	Australia	18 weeks at National Minimum Wage (currently A...	2 weeks at National Minimum Wage	Up to 52 weeks unpaid shared between the parents	Up to 3 weeks of unpaid leave	The 52 weeks are shared between the parents an...	52.0
3	Bahrain	60 days 100%	NaN	NaN	NaN	NaN	12.0
4	Bangladesh	16 weeks (8 weeks before delivery and 8 weeks ...	NaN	In case of third (+) time mother, who has two ...	NaN	NaN	18.0

In [19]:
```
ParentalLeave_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 40 entries, 0 to 40
Data columns (total 7 columns):
Country                 40 non-null object
Paid maternity leave    40 non-null object
Paid paternity leave    16 non-null object
Unpaid maternity leave  8 non-null object
Unpaid paternity leave  4 non-null object
Restrictions            7 non-null object
Maternity leave (weeks) 40 non-null float64
dtypes: float64(1), object(6)
memory usage: 3.8+ KB
```

Now I have 40 countries, and 40 numerical entries for Maternity Leave (weeks) in this table, so it can be easily merged with the other tables.

There is one last problem to address before merging the data from all the tables – some of the tables have non-numeric entries for Maternity Leave (weeks). I use the pandas .to_numeric() function to find the problem entries:

In [46]:
```
ParentalLeave_3.loc[pd.to_numeric(ParentalLeave_3['Maternity leave (weeks)'],errors='coerce').isnull()]
```

Out[46]:

	Country	Maternity leave (weeks)	Maternity leave (% of pay)	Paternity leave (weeks)	Paternity leave (% of pay)	Parental leave [For EITHER parent] (weeks)	Parental leave (% of pay)	Source of payment
35	Norway	35 (or 45)	100% for 25 weeks or 80% for 45 weeks	0-10 (depending on the mother's tax contributi...	100% or 80%	36 or 46 (10 for mothers; 10 for fathers; 26 t...	100% for 46 weeks or 80% for 56 weeks (up to a...	Social security
37	Portugal	17 (or 21)	100% for 17 weeks or 80% for 21	3	100%	13 each; "sharing bonus" of 4 weeks if initial...	25%	Social security
38	Romania	18 (9 weeks before the anticipated date of bir...	85%	5 days (15 days if an infant care course is ta...	100%	One parent is entitled to: 104 weeks (so until...	85%	Social security
44	Sweden	68 weeks or 480 days	80% (up to a ceiling)	18	80% (up to a ceiling)	60	80% (up to a ceiling) for 56 weeks; flat rate ...	Social security

Since there are only a few, they are easy to correct manually.

Now that each dataframe has a Country column and a Maternity Leave (weeks) column, I can easily concatenate them:

```
cols = ['Country','Maternity leave (weeks)']
MaternityLeave_all = \
pd.concat([ParentalLeave_0[cols], ParentalLeave_1[cols], ParentalLeave_2[cols], ParentalLeave_3[cols]])

# I will reset the index and also drop the old index:
MaternityLeave_all.reset_index(inplace=True)
MaternityLeave_all.drop('index', axis=1, inplace=True)
```

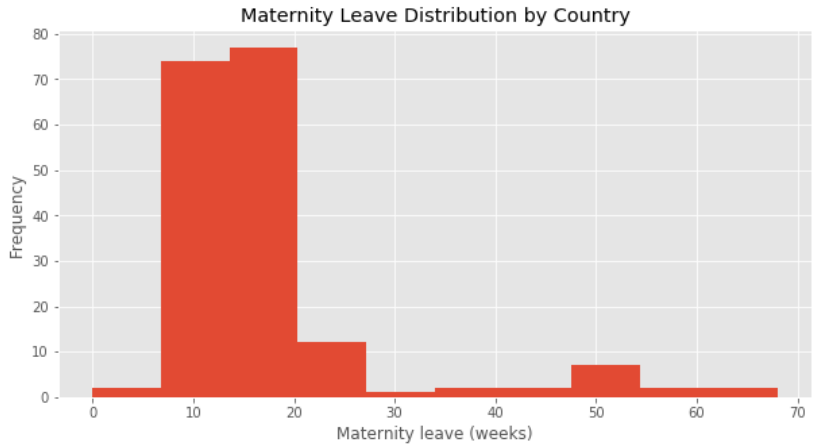
I wanted to take a look at some basic statistics within the Maternity Leave data, before comparing it to the employment data. This is easy to do, using some of pandas built in functions:

```
MaternityLeave_all['Maternity leave (weeks)'].describe()
```

```
count    181.000000
mean      17.604420
std       11.214236
min        0.000000
25%       12.000000
50%       14.000000
75%       18.000000
max       68.000000
Name: Maternity leave (weeks), dtype: float64
```

```
MaternityLeave_all['Maternity leave (weeks)'].hist(figsize=(10,5))
plt.title('Maternity Leave Distribution by Country')
plt.xlabel('Maternity leave (weeks)')
plt.ylabel('Frequency')
```

```
<matplotlib.text.Text at 0x10e56a550>
```



The distribution looks slightly bimodal – most of the data is concentrated around 18 weeks, but there is a smaller grouping of much longer maternity leaves, centered around 1 year.

Part IIb – Cleaning Employment Data

Next I need to clean up the employment data I downloaded from the OECD.

> Note: The OECD has amassed a huge amount of data. I downloaded the Full-time equivalend employment rate, but there are many more employment measures available on their site. I chose Full-time equivalent employment rate, because I thought it was a good measure of complete engagement in the workforce. It may be interesting to further investigate other measures of employment as a future project.

Again, I start by reading the csv into a pandas dataframe, printing out the first few rows, and using the describe() function to get a sense of what I was dealing with. This data a lot uglier than the parental leave tables from wikipedia. First of all, there are 1370 entries in the Country column, but only 39 unique values. This means countries are being repeated many times, which makes sense when you look at the other columns. Each row shows the employment rate for a certain country, for men or women, for a certain year. There are multiple rows for countries representing the employment rates for men and women for different years.

To narrow things down, I decided to look at only the results from 2014:

```
EmploymentRates_2014 = EmploymentRates.loc[EmploymentRates.Time==2014]
EmploymentRates_2014.describe(include='all')
```

My new dataframe, EmploymentRates_2014, looks a lot more manageable. Now there are 38 unique countries, and a total count of 76. This makes sense because each country has an entry for men and for women.

Next, I created a dataframe with three columns: Country, Men's Employment Rate, and Women's Employment Rate.

```
EmploymentRates_2014_CLEAN = pd.DataFrame({'Country':EmploymentRates_2014.Country.unique()})
EmploymentRates_2014_CLEAN['Employment Rate (Men)']=np.nan
EmploymentRates_2014_CLEAN['Employment Rate (Women)']=np.nan

for c in EmploymentRates_2014_CLEAN.Country:

    mens_employment_rate = \
        EmploymentRates_2014.loc[((EmploymentRates_2014.Country==c)&(EmploymentRates_2014.Sex=='Men')),'Value']
    womens_employment_rate = \
        EmploymentRates_2014.loc[((EmploymentRates_2014.Country==c)&(EmploymentRates_2014.Sex=='Women')),'Value']

    EmploymentRates_2014_CLEAN.loc[EmploymentRates_2014_CLEAN.Country==c,'Employment Rate (Men)'] = \
        mens_employment_rate.values[0]
    EmploymentRates_2014_CLEAN.loc[EmploymentRates_2014_CLEAN.Country==c, 'Employment Rate (Women)'] = \
        womens_employment_rate.values[0]

EmploymentRates_2014_CLEAN.head()
```

Part IIc – Merging Data and Generating Plots

Now that I would like to merge my cleaned up maternity leave and employment rate data and generate some plots to gain some insight into the relationship (or lack thereof) between the two. Since both dataframes have a column called Country, I can use pandas merge function:

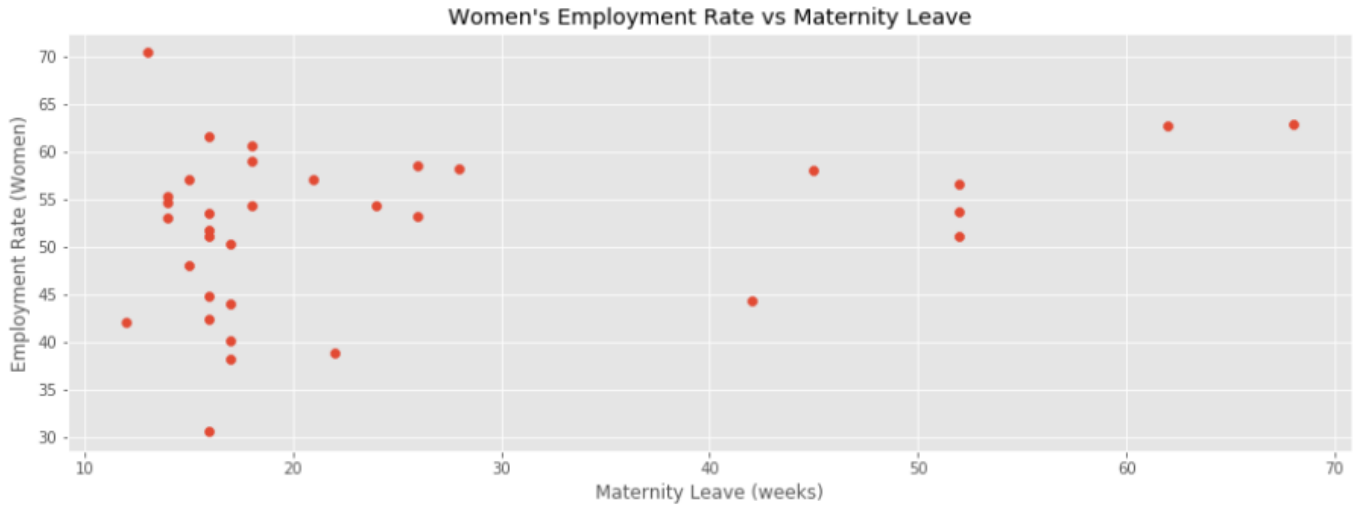
```
MaternityLeave_vs_Employment = MaternityLeave_all.merge(EmploymentRates_2014_CLEAN, on='Country')
```

Finally, after all that cleaning, it's time to make some plots!
I first did a straightforward plot of the length of maternity leave vs women's employment rates:

```
x = MaternityLeave_vs_Employment['Maternity leave (weeks)']
y = MaternityLeave_vs_Employment['Employment Rate (Women)']

plt.figure(figsize=(15,5))
plt.scatter(x,y)

plt.title('Women\'s Employment Rate vs Maternity Leave')
plt.xlabel('Maternity Leave (weeks)')
plt.ylabel('Employment Rate (Women)')
```



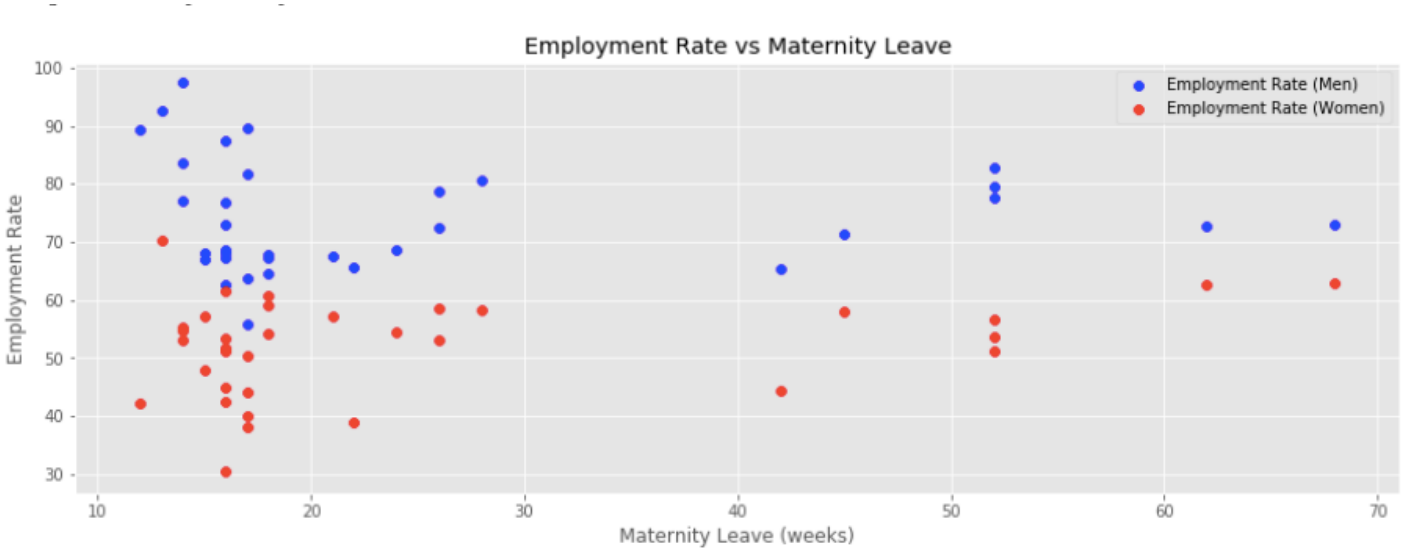
I was surprised to see that some of the countries with the longest maternity leaves also have pretty high employment rates for women. Perhaps this is related more to the overall strength of the economy. I wanted to plot men's employment rates too, to see if there is the same sort of pattern.

```
x = MaternityLeave_vs_Employment['Maternity leave (weeks)']
y1 = MaternityLeave_vs_Employment['Employment Rate (Men)']
y2 = MaternityLeave_vs_Employment['Employment Rate (Women)']

plt.figure(figsize=(15,5))
plt.scatter(x,y1, color='blue')

plt.scatter(x,y2, color='red')

plt.title('Employment Rate vs Maternity Leave')
plt.xlabel('Maternity Leave (weeks)')
plt.ylabel('Employment Rate')
plt.legend()
```



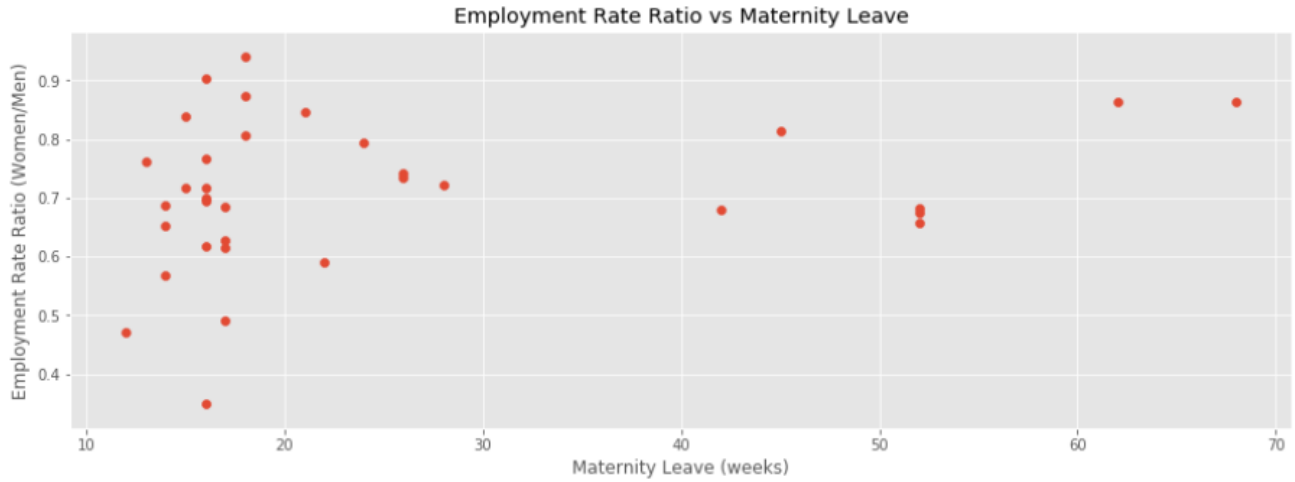
It looks like the countries with maternity leave over 40 weeks have a stronger relationship between men's and women's employment rates that the countries with shorter maternity leaves. This suggests that the employment rates in those countries are more affected by the overall economy than the maternity leave policies. The countries with under 30 weeks seem to have a less strong relationship.

Another way to account for the overall economy of a country would be to plot the **ratio** of women's to men's employment. This is more of a measure of equal participation in the workforce.

I first added a new column to my dataframe to represent the ratio of women's to men's employment:

```
MaternityLeave_vs_Employment['Ratio of Women\'s to Men\'s Employment Rate'] = \
    MaternityLeave_vs_Employment['Employment Rate (Women)']/MaternityLeave_vs_Employment['Employment Rate (Men)']
```

Next I plot the ratio of women's to men's employment vs the number of weeks of maternity leave:



Interestingly, there seems to be a peak around 18 weeks! I think there is a bit more digging to do before advising countries to adopt a rule of 18 week maternity leaves to optimize equality in the workplace. Some of the countries with very long maternity leaves have surprisingly high ratios of women's to men's employment rates. I also came up with a few more questions to think about …

Questions for Further Investigation

While I was working through this project, I came up with a few more detailed questions for further investigation:

- Is there any difference in women's engagement in the workforce in the workforce in countries with more equal parental leave for both parents (not just available to mothers or "primary caregivers")?
- How is parental leave funded? If it is paid for by public funds rather than by the hiring company, are companies more willing to hire women of childbearing age?
- Is there a difference in employment between countries with paid and unpaid parental leave?
- How does all of this relate to the fertility rate? Are women more likely to have more children when there is more maternity leave, or less likely to work altogether in countries with higher fertility rates?