idss_lecture_01_introduction_demo

September 20, 2022

[1]:
```
%%javascript
$('#menubar').toggle();
```

```
<IPython.core.display.Javascript object>
```

1 Introduction to Data Science and Systems 2022-2023

1.1 Lecture Week 1: Introduction - *a basic data science example*

University of Glasgow v20222023a - *Example adapted from A. Joseph and the DS100 textbook.*

1.2 Load the basic packages…

[2]:
```
!pip install plotly
```

```
Requirement already satisfied: plotly in
/home/nicolas/anaconda3/lib/python3.9/site-packages (5.6.0)
Requirement already satisfied: tenacity>=6.2.0 in
/home/nicolas/anaconda3/lib/python3.9/site-packages (from plotly) (8.0.1)
Requirement already satisfied: six in
/home/nicolas/anaconda3/lib/python3.9/site-packages (from plotly) (1.16.0)
```

[3]:
```
# You do not need to understand all of this in detail yet
# - this notebook is just for inspiration and motivation

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns # (this is a fancy plotting libary - we will mostly be␣
↪using matplotlib in this course)

## Plotly plotting support (this is a fancy plotting libary - we will mostly be␣
↪using matplotlib in this course)
import plotly.offline as py
py.init_notebook_mode()
import plotly.graph_objs as go
```

```python
import plotly.figure_factory as ff
import plotly.express as px
```

1.3 1) The questions

- How many students in IDSS this semester ?
- What is the distribution of the degree programmes in IDSS this year?
- Key questions: What is the gender distribution in IDSS this semester?

1.4 2) Data acquisition and availability

- You provided some of the data when registering via MyCampus. The data was thus collected for general purpose administrating and not necessarily to answer your specific question.

1.5 3) Storage and access

- MyCampus (Oracle)
- SoCS LTC system (SQL)
- Moodle (activity)

Some observations: - The data based contain sensitive information - we need to make sure the data is safely and can only be accessed by approved users (and inline with GDPR)

---

2 4) Data exploration, preparation and curation

2.1 4.1) Loading and inspecting the data using Pandas

We need a data structure to load the data into for visualisation, querying and exploration.

Here we we use Pandas because they provide built-in functionality to easily explore the (see Moodle for suggested study material)

You can read more about panda here https://pandas.pydata.org/

[4]:
```python
data = pd.read_csv("roster_mod_20202021c.csv")
len(data)
```

[4]:
```
374
```

[5]:
```python
data.head(20)
```

[5]:
```
   Level ProgrammeID  Firstname                  Advisor  \
0     UG         NaN     Andrew                Rogers,S
1    PGT   I261-5200      Mitko                Rogers,S
```

```
2    PGT   I261-5200        Euan                Rogers,S
3    PGT   G511-5200        John                Rogers,S
4    PGT   I261-5200       Lucas                Rogers,S
5    PGT   G577-5200       Grant  ChiefAdviser-Science,O
6    PGT   G577-5200  Kleanthis                 Rogers,S
7    PGT   I261-5200     Charles                Rogers,S
8    PGT   I261-5200        Xiao                Rogers,S
9    PGT   I261-5200     Pengjun                Rogers,S
10   PGT   I261-5200      Qiling                Rogers,S
11   PGT   I261-5200        Yuli                Rogers,S
12   PGT   G511-5200       Jihua                Rogers,S
13   PGT   I261-5200    Mengting  ChiefAdviser-Science,O
14   PGT   I261-5200      Yuchen                Rogers,S
15   PGT   G577-5200      Zhehao                Rogers,S
16   PGT   I261-5200      Zitong                Rogers,S
17   PGT   I261-5200    Guanxuan  ChiefAdviser-Science,O
18   PGT   I261-5200    YINGHONG                Rogers,S
19   PGT   G511-5200    Mohammad                Rogers,S

                    Programme Gender
0      Computer Science, BSc      M
1        MSc in Data Science    NaN
2        MSc in Data Science    NaN
3      Computing Science,MSc    NaN
4        MSc in Data Science    NaN
5   Information Security,MSc    NaN
6   Information Security,MSc    NaN
7        MSc in Data Science    NaN
8        MSc in Data Science    NaN
9        MSc in Data Science    NaN
10       MSc in Data Science    NaN
11       MSc in Data Science    NaN
12     Computing Science,MSc    NaN
13       MSc in Data Science    NaN
14       MSc in Data Science    NaN
15  Information Security,MSc    NaN
16       MSc in Data Science    NaN
17       MSc in Data Science    NaN
18       MSc in Data Science    NaN
19     Computing Science,MSc    NaN
```

[6]:
```
data['Firstname'].unique()
len(data['Firstname'].unique())
```

[6]: 349

2.2 4.2) Curation and cleaning

[7]:
```
data['Firstname'] = data['Firstname'].str.lower()
print("Number of Students:", len(data))
data.head(20)
```

```
Number of Students: 374
```

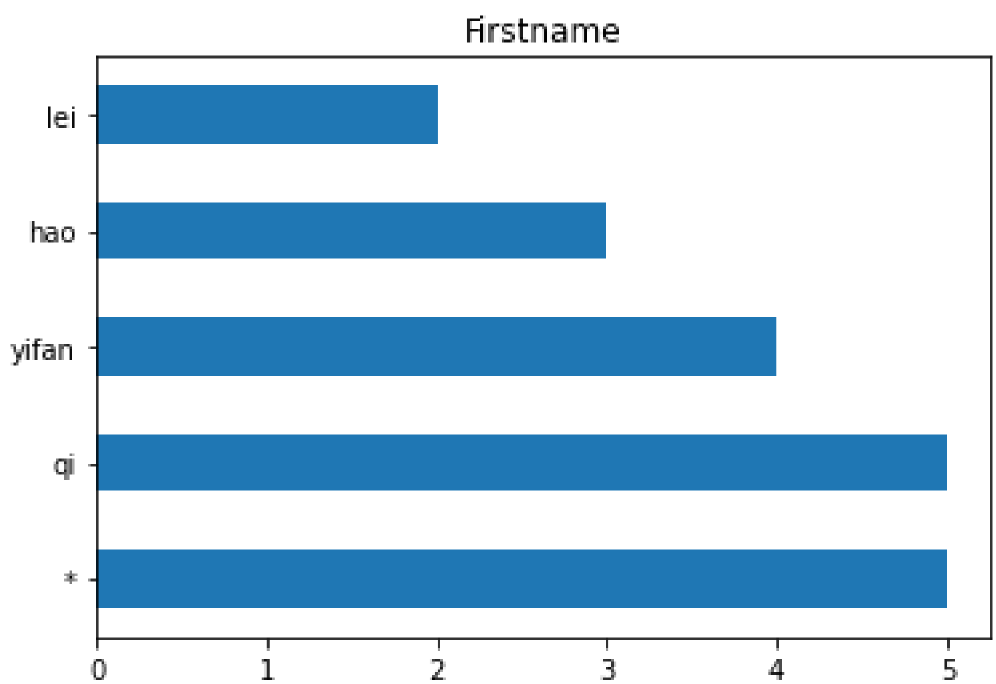
[7]:
```
   Level ProgrammeID  Firstname                   Advisor  \
0     UG         NaN     andrew                 Rogers,S
1    PGT   I261-5200      mitko                 Rogers,S
2    PGT   I261-5200       euan                 Rogers,S
3    PGT   G511-5200       john                 Rogers,S
4    PGT   I261-5200      lucas                 Rogers,S
5    PGT   G577-5200      grant  ChiefAdviser-Science,O
6    PGT   G577-5200  kleanthis                 Rogers,S
7    PGT   I261-5200    charles                 Rogers,S
8    PGT   I261-5200       xiao                 Rogers,S
9    PGT   I261-5200    pengjun                 Rogers,S
10   PGT   I261-5200     qiling                 Rogers,S
11   PGT   I261-5200       yuli                 Rogers,S
12   PGT   G511-5200      jihua                 Rogers,S
13   PGT   I261-5200   mengting  ChiefAdviser-Science,O
14   PGT   I261-5200     yuchen                 Rogers,S
15   PGT   G577-5200     zhehao                 Rogers,S
16   PGT   I261-5200     zitong                 Rogers,S
17   PGT   I261-5200   guanxuan  ChiefAdviser-Science,O
18   PGT   I261-5200   yinghong                 Rogers,S
19   PGT   G511-5200   mohammad                 Rogers,S

                  Programme Gender
0      Computer Science, BSc      M
1        MSc in Data Science    NaN
2        MSc in Data Science    NaN
3      Computing Science,MSc    NaN
4        MSc in Data Science    NaN
5   Information Security,MSc    NaN
6   Information Security,MSc    NaN
7        MSc in Data Science    NaN
8        MSc in Data Science    NaN
9        MSc in Data Science    NaN
10       MSc in Data Science    NaN
11       MSc in Data Science    NaN
12     Computing Science,MSc    NaN
13       MSc in Data Science    NaN
14       MSc in Data Science    NaN
15  Information Security,MSc    NaN
```

```
16        MSc in Data Science     NaN
17        MSc in Data Science     NaN
18        MSc in Data Science     NaN
19      Computing Science,MSc     NaN
```

[8]:
```
data['Firstname'].unique()
len(data['Firstname'].unique())
```

[8]: 343

[9]:
```
(
    data["Firstname"]
        .str.lower()
        .value_counts().sort_values(ascending=False)
        .head(5).plot(kind='barh', title = "Firstname")
);
```



[10]:
```
data = data.drop(data[data.Programme == "Computer Science, BSc"].index)
len(data)
```

[10]: 372

2.3 4.3) Basic visualization and summarization

[11]:
```
data.describe()
```

[11]:
```
        Level ProgrammeID Firstname   Advisor             Programme Gender
count     372         372       372       372                   370      6
unique      1           3       341        12                     3      2
top       PGT   I261-5200         *  Rogers,S  MSc in Data Science      F
freq      372         190         5        56                   188      4
```

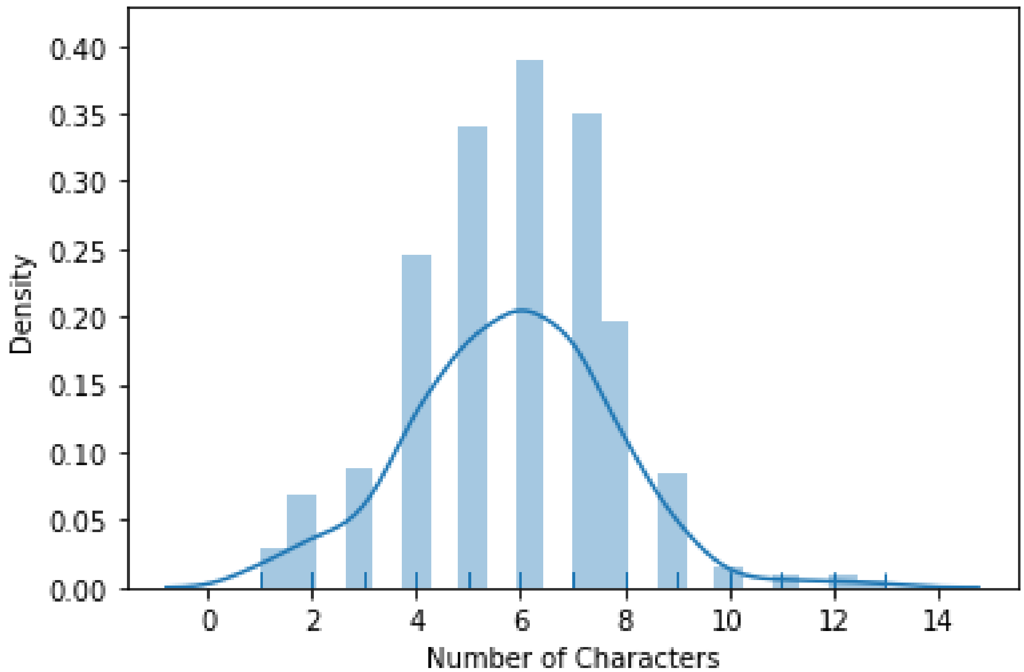
---

3 5) Data modelling and analysis

3.1 Q: What is the distribution of the lengths of students' names in this class?

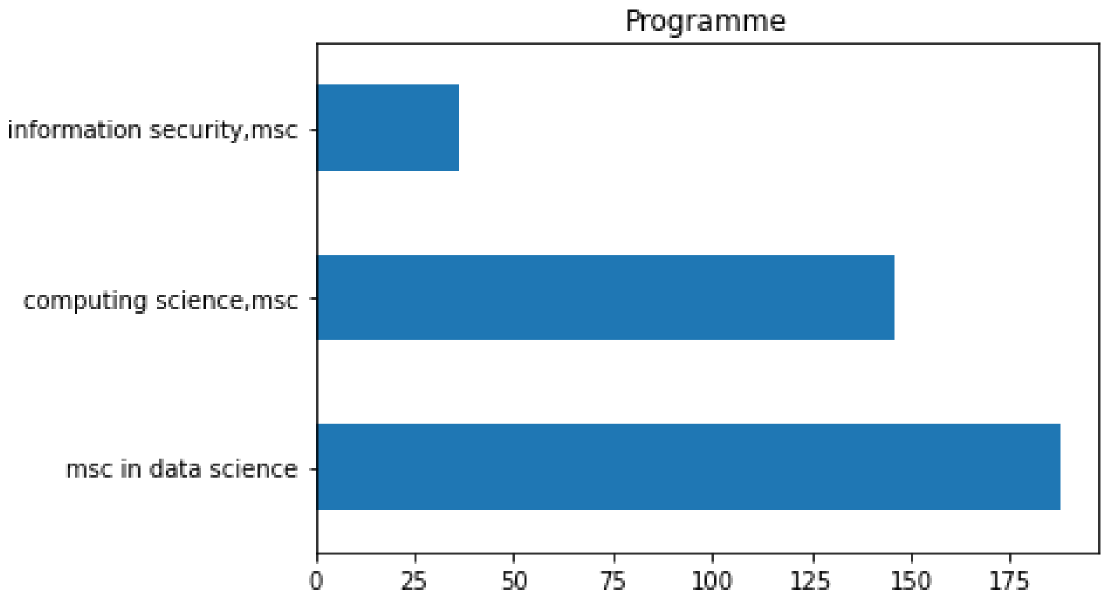
[47]:
```
sns.distplot(data['Firstname'].str.len(), rug=True, axlabel="Number of␣
↪Characters");
```

```
/home/nicolas/anaconda3/lib/python3.9/site-
packages/seaborn/distributions.py:2619: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version.
Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

/home/nicolas/anaconda3/lib/python3.9/site-
packages/seaborn/distributions.py:2103: FutureWarning:

The `axis` variable is no longer used and will be removed. Instead, assign
variables directly to `x` or `y`.
```

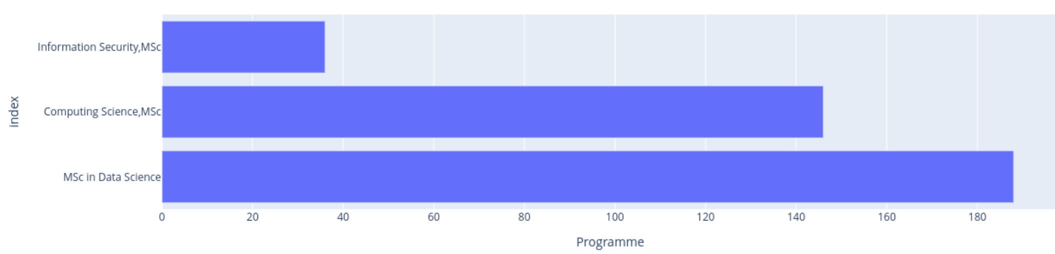


3.2 Q: What are the programme degrees of students in the class?

[13]:
```
(
    data["Programme"]
        .str.lower()
        .value_counts().sort_values(ascending=False)
        .head(20).plot(kind='barh', title = "Programme")
);
```



[14]:
```
# fancy plotting using ploty (we won't actually use it the course; is only for
↪inspiration)
px.bar(data['Programme'].value_counts().to_frame().reset_index().head(20),
       x = 'Programme',
       y = 'index',
       orientation = 'h')
```



3.3 Q: What is the gender distribution of the class?

Can we answer this questions with the raw data?

[15]:
```
print(data.columns)
```

```
Index(['Level', 'ProgrammeID', 'Firstname', 'Advisor', 'Programme', 'Gender'],
dtype='object')
```

So the answer is maybe as we do have a collum called Gender, but remember all the NaNs...

[16]:
```python
data.head(20)
```

[16]:
```
   Level ProgrammeID  Firstname                    Advisor  \
1    PGT   I261-5200      mitko                   Rogers,S
2    PGT   I261-5200       euan                   Rogers,S
3    PGT   G511-5200       john                   Rogers,S
4    PGT   I261-5200      lucas                   Rogers,S
5    PGT   G577-5200      grant  ChiefAdviser-Science,O
6    PGT   G577-5200  kleanthis                   Rogers,S
7    PGT   I261-5200    charles                   Rogers,S
8    PGT   I261-5200       xiao                   Rogers,S
9    PGT   I261-5200    pengjun                   Rogers,S
10   PGT   I261-5200     qiling                   Rogers,S
11   PGT   I261-5200       yuli                   Rogers,S
12   PGT   G511-5200      jihua                   Rogers,S
13   PGT   I261-5200   mengting  ChiefAdviser-Science,O
14   PGT   I261-5200     yuchen                   Rogers,S
15   PGT   G577-5200     zhehao                   Rogers,S
16   PGT   I261-5200     zitong                   Rogers,S
17   PGT   I261-5200   guanxuan  ChiefAdviser-Science,O
18   PGT   I261-5200   yinghong                   Rogers,S
19   PGT   G511-5200   mohammad                   Rogers,S
20   PGT   I261-5200     yuzhou                   Rogers,S

                    Programme Gender
1        MSc in Data Science    NaN
2        MSc in Data Science    NaN
3      Computing Science,MSc    NaN
4        MSc in Data Science    NaN
5   Information Security,MSc    NaN
6   Information Security,MSc    NaN
7        MSc in Data Science    NaN
8        MSc in Data Science    NaN
9        MSc in Data Science    NaN
10       MSc in Data Science    NaN
11       MSc in Data Science    NaN
12     Computing Science,MSc    NaN
13       MSc in Data Science    NaN
14       MSc in Data Science    NaN
15  Information Security,MSc    NaN
16       MSc in Data Science    NaN
17       MSc in Data Science    NaN
18       MSc in Data Science    NaN
```

```
19     Computing Science,MSc    NaN
20       MSc in Data Science    NaN
```

Ideas:

- What do we mean by gender?
- Can we use the name to estimate gender?
- How would we build model of gender given the name?
- Where can we get data for such a model?

3.3.1 Build a model based on an aux dataset

- Easily accessible data is from the US SSA (https://www.ssa.gov/oact/babynames)? ***Is it going to case problem that we are using US data in the UK?***

[17]:
```
import urllib.request
import os.path

# Download data from the web directly
data_url = "https://www.ssa.gov/oact/babynames/names.zip"
local_filename = "babynames.zip"
if not os.path.exists(local_filename): # if the data exists don't download again
    with urllib.request.urlopen(data_url) as resp, open(local_filename, 'wb')␣
↪as f:
        f.write(resp.read())
```

[18]:
```
# Load data without unzipping the file
import zipfile
babynames = []
with zipfile.ZipFile(local_filename, "r") as zf:
    data_files = [f for f in zf.filelist if f.filename[-3:] == "txt"]
    def extract_year_from_filename(fn):
        return int(fn[3:7])
    for f in data_files:
        year = extract_year_from_filename(f.filename)
        with zf.open(f) as fp:
            df = pd.read_csv(fp, names=["Name", "Sex", "Count"])
            df["Year"] = year
            babynames.append(df)
babynames = pd.concat(babynames)


babynames.head()
```

[18]:

	Name	Sex	Count	Year
0	Mary	F	7065	1880
1	Anna	F	2604	1880
2	Emma	F	2003	1880

```
3  Elizabeth    F    1939  1880
4     Minnie    F    1746  1880
```

A bit of data clearning

[19]:
```
babynames['Name'] = babynames['Name'].str.lower()
babynames.tail()
```

[19]:
```
          Name Sex  Count  Year
31949  zyheem   M      5  2019
31950   zykel   M      5  2019
31951  zyking   M      5  2019
31952     zyn   M      5  2019
31953   zyran   M      5  2019
```

How many people does this data represent?

[20]:
```
format(babynames['Count'].sum(), ',d')
```

[20]:
```
'355,149,899'
```

[21]:
```
len(babynames)
```

[21]:
```
1989401
```

Is this number low or high?

It seems low. However the social security website states:

All names are from Social Security card applications for births that occurred in the United States after 1879. **Note that many people born before 1937 never applied for a Social Security card, so their names are not included in our data.** For others who did apply, our records may not show the place of birth, and again their names are not included in our data. All data are from a 100% sample of our records on Social Security card applications as of the end of February 2016.

Let's query to find rows that match desired conditions.

[22]:
```
babynames[(babynames['Name'] == 'vela') & (babynames['Sex'] == 'F')].tail(5)
```

[22]:
```
      Name Sex  Count  Year
6013  vela   F     22  2015
5594  vela   F     24  2016
7405  vela   F     16  2017
6213  vela   F     20  2018
6638  vela   F     18  2019
```

[23]:
```
babynames[(babynames['Name'] == 'anthony') & (babynames['Year'] == 2000)]
```

[23]:
```
            Name Sex  Count  Year
2782   anthony   F     52  2000
17673  anthony   M  19652  2000
```

[25]:
```python
babynames.query('Name.str.contains("data")', engine='python')
```

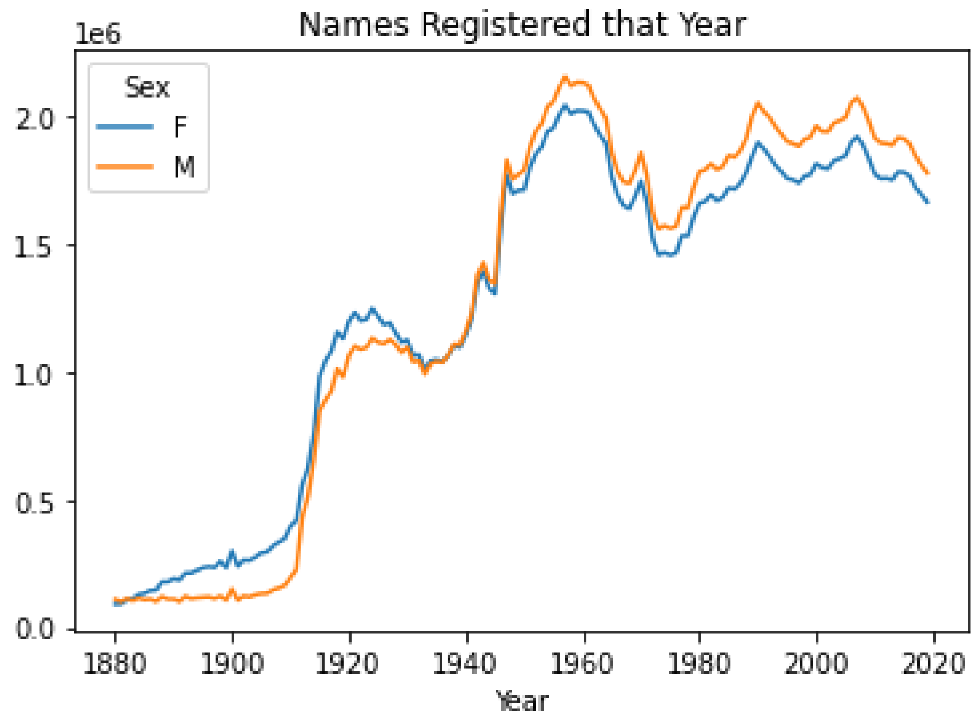
[25]:
```
              Name Sex  Count  Year
9762        kidata   F      5  1975
24914     datavion   M      5  1995
23610   datavious   M      7  1997
12102      datavia   F      7  2000
27507     datavion   M      6  2001
28910       datari   M      5  2001
29138     datavian   M      5  2002
29139   datavious   M      5  2002
30572     datavion   M      5  2004
17139      datavia   F      5  2005
31027     datavion   M      5  2005
31021     datavion   M      6  2006
33338   datavious   M      5  2007
33339     datavius   M      5  2007
33402   datavious   M      5  2008
33081     datavion   M      5  2009
32497   datavious   M      5  2010
```

Subquestion: What is the proportion of Male and Female Individuals Over Time? In this example we construct a pivot table which aggregates the number of babies registered for each year by Sex.

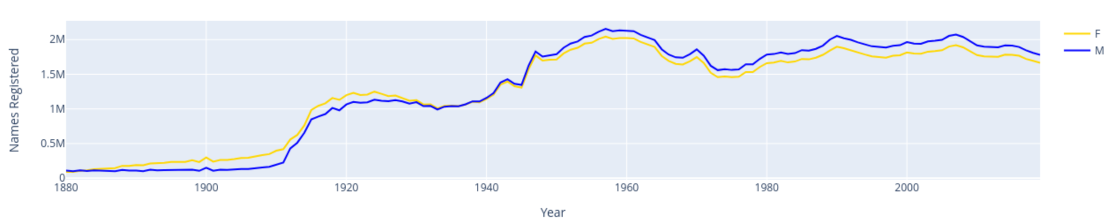
[26]:
```python
pivot_year_name_count = pd.pivot_table(babynames,
        index=['Year'], # the row index
        columns=['Sex'], # the column values
        values='Count', # the field(s) to processed in each group
        aggfunc=np.sum,
    )

pivot_year_name_count.head()
```

[26]:
```
Sex        F       M
Year
1880   90994  110490
1881   91953  100743
1882  107847  113686
1883  112319  104625
1884  129019  114442
```

[27]:
```python
pivot_year_name_count.plot(title='Names Registered that Year');
```

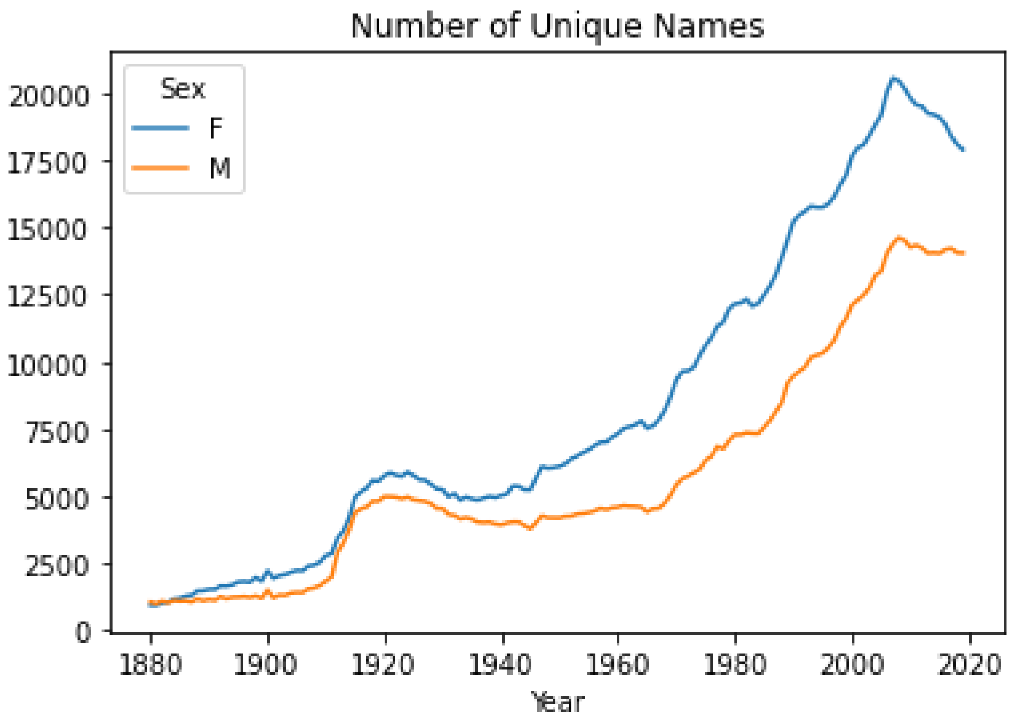


[28]:
```
fig = go.Figure()
fig.add_trace(go.Scatter(x = pivot_year_name_count.index, y =␣
 ↪pivot_year_name_count['F'], name = 'F', line=dict(color='gold')))
fig.add_trace(go.Scatter(x = pivot_year_name_count.index, y =␣
 ↪pivot_year_name_count['M'], name = 'M', line=dict(color='blue')))
fig.update_layout(xaxis_title = 'Year', yaxis_title = 'Names Registered')
```



3.3.2 Subquestion: How many unique names for each year?

[29]:
```
pivot_year_name_nunique = pd.pivot_table(babynames,
        index=['Year'],
        columns=['Sex'],
        values='Name',
        aggfunc=lambda x: len(np.unique(x)),
    )

pivot_year_name_nunique.plot(
   title='Number of Unique Names');
```



Some observations: - Registration data seems limited in the early 1900s. Because many people did not register before 1937. - You can see the baby boomers and the echo boom. - Females have greater diversity of names.

3.3.3 Subquestion: Computing the Proportion of Female Babies For Each Name:

[30]:
```
sex_counts = pd.pivot_table(babynames, index='Name', columns='Sex',␣
↪values='Count',
                            aggfunc='sum', fill_value=0., margins=True)
sex_counts.head()
```

[30]:
```
Sex        F    M  All
Name
aaban      0  120  120
aabha     40    0   40
aabid      0   16   16
aabidah    5    0    5
aabir      0   10   10
```

Compute proportion of female babies given each name.

[31]:
```
prop_female = sex_counts['F'] / sex_counts['All']
prop_female.head(10)
```

[31]:
```
Name
aaban        0.0
aabha        1.0
aabid        0.0
aabidah      1.0
aabir        0.0
aabriella    1.0
aada         1.0
aadam        0.0
aadan        0.0
aadarsh      0.0
dtype: float64
```

[32]:
```
prop_female.tail(10)
```

[32]:
```
Name
zytavion     0.000000
zytavious    0.000000
zyus         0.000000
zyva         1.000000
zyvion       0.000000
zyvon        0.000000
zyyanna      1.000000
zyyon        0.000000
zzyzx        0.000000
All          0.494913
dtype: float64
```

Testing a few names

[33]:
```
prop_female['audi']
```

[33]:
```
0.5978260869565217
```

[34]:
```
prop_female['anthony']
```

[34]: 0.004856689867035234

[35]:
```
prop_female['joey']
```

[35]: 0.1133165658350894

[36]:
```
prop_female['mark']
```

[36]: 0.003307100877990732

[37]:
```
prop_female["sarah"]
```

[37]: 0.9969322438050136

[38]:
```
prop_female["min"]
```

[38]: 0.37598736176935227

[39]:
```
prop_female["pat"]
```

[39]: 0.600140600694029

3.3.4 Modelling

Idea: Build Simple Classifier (Model) based on lookup table.

We can define a function to return the most likely Sex for a name. If there is an exact tie, the function returns Male. If the name does not appear in the social security dataset, we return Unknown.

[40]:
```
def sex_from_name(name):
    lower_name = name.lower()
    if lower_name in prop_female.index:
        return 'F' if prop_female[lower_name] > 0.5 else 'M'
    else:
        return "Unknown"
```

[41]:
```
sex_from_name("audi")
```

[41]: 'F'

[42]:
```
sex_from_name("joey")
```

[42]: 'M'

What fraction of students in IDSS this semester have names in the SSN dataset?

[43]:
```
student_names = pd.Index(data["Firstname"]).intersection(prop_female.index)
print("Fraction of names in the babynames data:" , len(student_names) /
↪len(data))
```

```
Fraction of names in the babynames data: 0.3655913978494624
```

Which names are not in the dataset? Why might these names not appear?

[44]:
```
missing_names = pd.Index(data["Firstname"]).difference(prop_female.index)
print(missing_names)
print(len(missing_names))
```

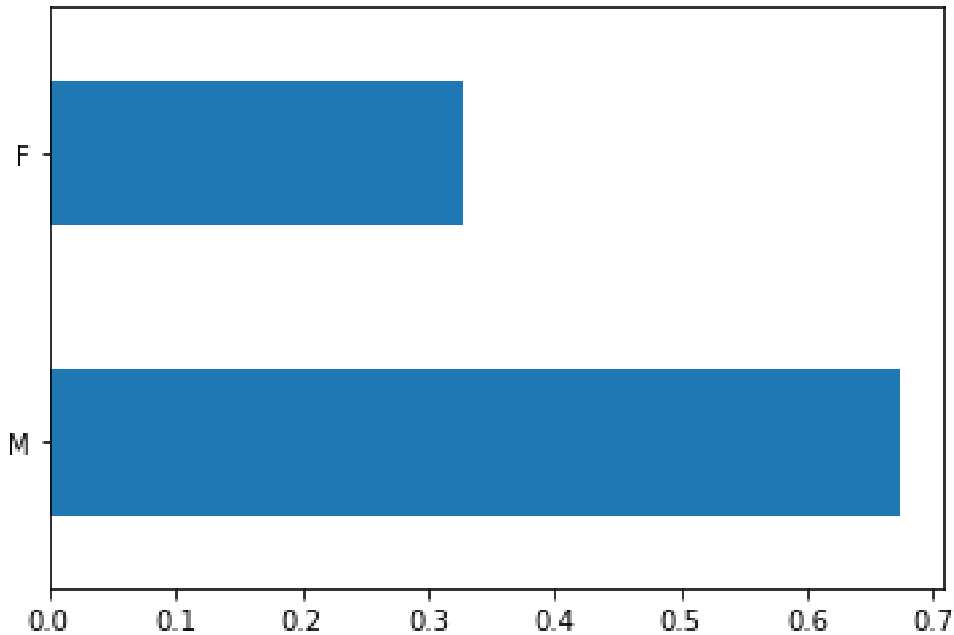
```
Index(['*', '-', 'aikaterini', 'anhua', 'avinab', 'botao', 'boyang', 'boyuan',
       'buruo', 'ceyu',
       …
       'zhehao', 'zhenyu', 'zhihan', 'zhoutian', 'zhu', 'zhuanghai', 'zhuo',
       'zidi', 'zijia', 'zixia'],
      dtype='object', length=205)
205
```

Observation: - That seems like a lot of missing names! - Should we continue with this dataset or find a better data source?

3.4 Q: What is the fraction of female and male students?

Back to the orginal and final question?

[45]:
```
data['Pred. Sex'] = data['Firstname'].apply(sex_from_name)
(data[data['Pred. Sex'] != "Unknown"]['Pred. Sex'].value_counts()/
↪len(data[data['Pred. Sex'] != "Unknown"])).plot(kind="barh");
```



3.5 6) Evaluation

- ***Is this an accurate result ? Let's do an post-hoc survey and find out!***

Additionally: - Which visualisations do you need to communicate your findings? - How can you concisely describe the data, process, results and conclusions. - How can you summarise and document the steps you have taken? - What are the limitations of the analysis? … should we rerun the whole analysis with a different strategy and perhaps data datasets?