CAPSTONE PROJECT 1 - DATA WRANGLING

INTRODUCTION

An explorative analysis on the country and states in the US affected by the Deferred Action for Childhood Arrivals (DACA). To study and analyze the impact of DACA and recent trends as per the last four quarters in 2017.

HOW DATA IS COLLECTED?

- Data will be collected from the USCIS link for the 2017 and 2016.
- All data will be converted from PDF to excel data.
- Data will be divided into four parts – Case status, Country Status, States Status and 2017 Status

DATA SOURCE

https://www.uscis.gov/tools/reports-studies/immigration-forms-data/data-set-form-i-821d-deferred-action-childhood-arrivals

COMMON PROBLEMS FOUND IN DATASETS

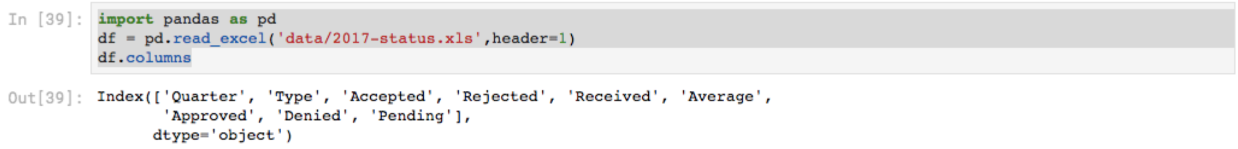
- Inconsistent column names
- Missing Data
- Outliners
- Duplicate rows
- Untidy
- Need to process columns
- Column type signal unexpected data values

Let’s go over each problem and what I did to wrangle the data.

INCONSISTENT COLUMN NAMES

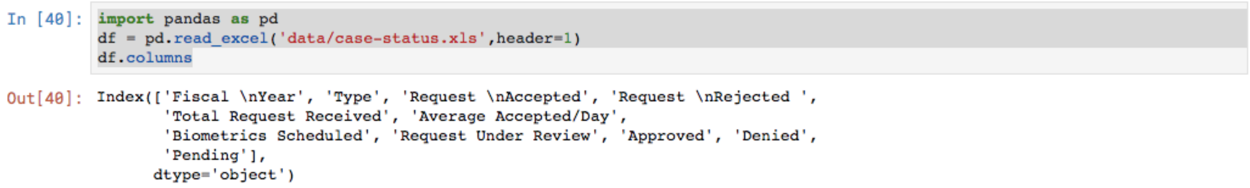
Downloaded the pdf from the USCIS site and convert that into excel sheet using an online tool. Now the headings are inconsistent with title case, uppercase and some space. Load the data into pandas and checked how inconsistent the column headers using **columns().** All examples shown below.

2017-status.xls



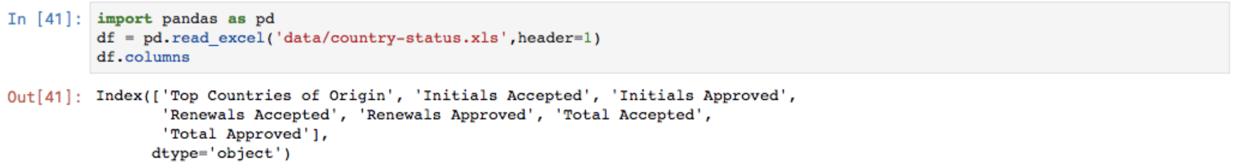
```
In [39]: import pandas as pd
         df = pd.read_excel('data/2017-status.xls',header=1)
         df.columns

Out[39]: Index(['Quarter', 'Type', 'Accepted', 'Rejected', 'Received', 'Average',
                'Approved', 'Denied', 'Pending'],
               dtype='object')
```

case-status.xls



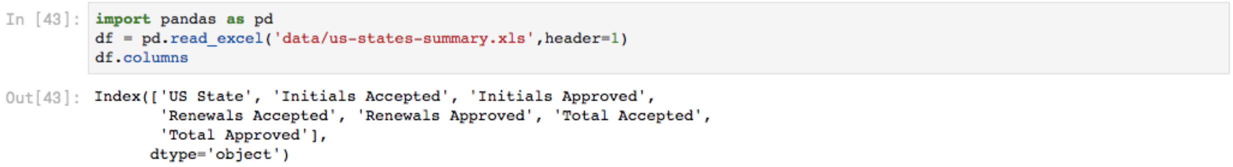
```
In [40]: import pandas as pd
         df = pd.read_excel('data/case-status.xls',header=1)
         df.columns

Out[40]: Index(['Fiscal \nYear', 'Type', 'Request \nAccepted', 'Request \nRejected ',
                'Total Request Received', 'Average Accepted/Day',
                'Biometrics Scheduled', 'Request Under Review', 'Approved', 'Denied',
                'Pending'],
               dtype='object')
```

country-status.xls



```
In [41]: import pandas as pd
         df = pd.read_excel('data/country-status.xls',header=1)
         df.columns

Out[41]: Index(['Top Countries of Origin', 'Initials Accepted', 'Initials Approved',
                'Renewals Accepted', 'Renewals Approved', 'Total Accepted',
                'Total Approved'],
               dtype='object')
```

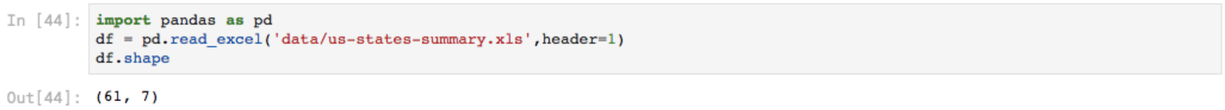
us-states-summary.xls



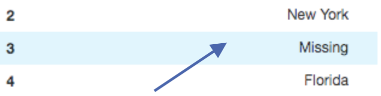
```
In [43]: import pandas as pd
         df = pd.read_excel('data/us-states-summary.xls',header=1)
         df.columns

Out[43]: Index(['US State', 'Initials Accepted', 'Initials Approved',
                'Renewals Accepted', 'Renewals Approved', 'Total Accepted',
                'Total Approved'],
               dtype='object')
```

MISSING DATA

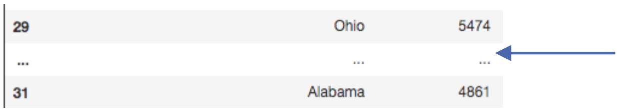
Using shape() functionality checked the states summary to see whether any data is missing. As you see below the number of states returned is 61 with 7 columns. On investigating further found out there was some null rows, one row with state name as 'missing' and few other rows with values which are not states of US but considered as region.



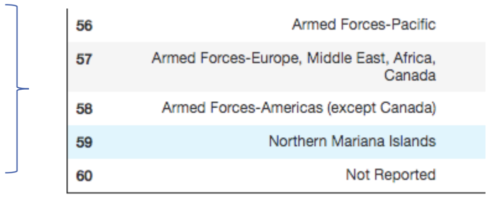
```
In [44]: import pandas as pd
         df = pd.read_excel('data/us-states-summary.xls',header=1)
         df.shape

Out[44]: (61, 7)
```



2	New York
3	Missing
4	Florida



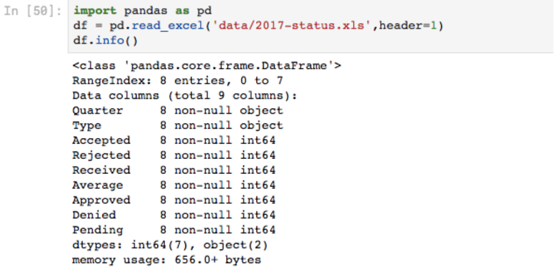
29	Ohio	5474
...	...	...
31	Alabama	4861



56	Armed Forces-Pacific
57	Armed Forces-Europe, Middle East, Africa, Canada
58	Armed Forces-Americas (except Canada)
59	Northern Mariana Islands
60	Not Reported

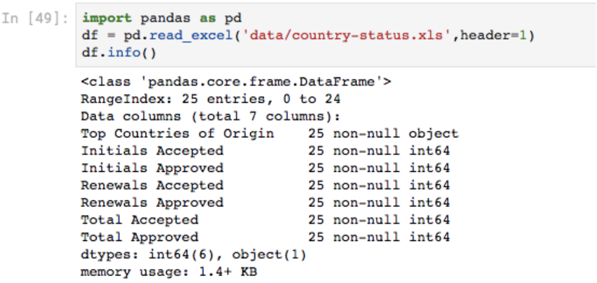
CHECKING DATA TYPES FOR ALL DATA

Using info() to get additional information about each dataset. On examining the results found that case-status datasets has total of 10 rows but column Biometrics scheduled and request under review had only 2 values. So there were 8 missing values. Also in the same data sets noticed that these two columns are of datatype float64 and Denied column is object data type which will be treated like string. Denied column should have been int64 datatype.



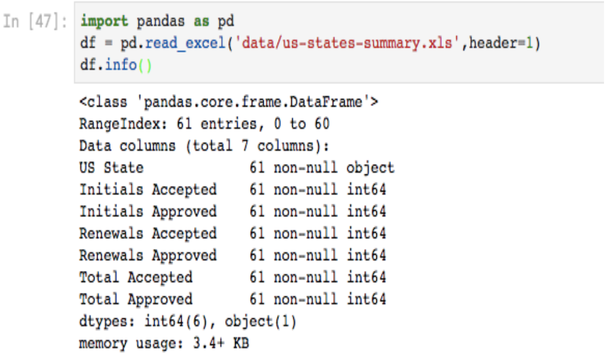
```
In [50]: import pandas as pd
         df = pd.read_excel('data/2017-status.xls',header=1)
         df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8 entries, 0 to 7
Data columns (total 9 columns):
Quarter     8 non-null object
Type        8 non-null object
Accepted    8 non-null int64
Rejected    8 non-null int64
Received    8 non-null int64
Average     8 non-null int64
Approved    8 non-null int64
Denied      8 non-null int64
Pending     8 non-null int64
dtypes: int64(7), object(2)
memory usage: 656.0+ bytes
```



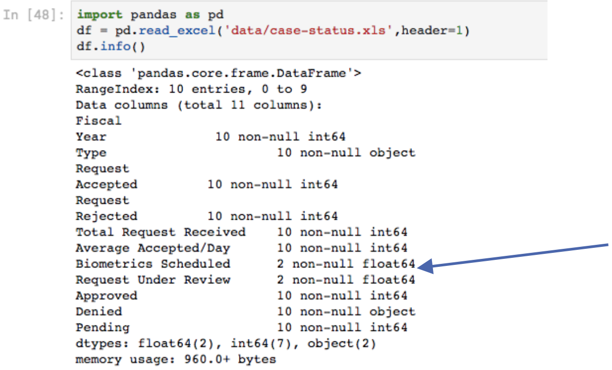
```
In [49]: import pandas as pd
         df = pd.read_excel('data/country-status.xls',header=1)
         df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 7 columns):
Top Countries of Origin    25 non-null object
Initials Accepted          25 non-null int64
Initials Approved          25 non-null int64
Renewals Accepted          25 non-null int64
Renewals Approved          25 non-null int64
Total Accepted             25 non-null int64
Total Approved             25 non-null int64
dtypes: int64(6), object(1)
memory usage: 1.4+ KB
```



```
In [47]: import pandas as pd
         df = pd.read_excel('data/us-states-summary.xls',header=1)
         df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 61 entries, 0 to 60
Data columns (total 7 columns):
US State             61 non-null object
Initials Accepted    61 non-null int64
Initials Approved    61 non-null int64
Renewals Accepted    61 non-null int64
Renewals Approved    61 non-null int64
Total Accepted       61 non-null int64
Total Approved       61 non-null int64
dtypes: int64(6), object(1)
memory usage: 3.4+ KB
```

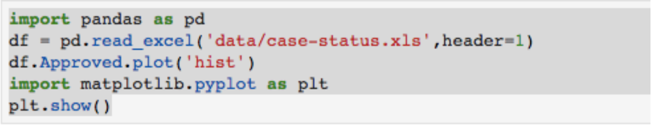


```
In [48]: import pandas as pd
         df = pd.read_excel('data/case-status.xls',header=1)
         df.info()
```

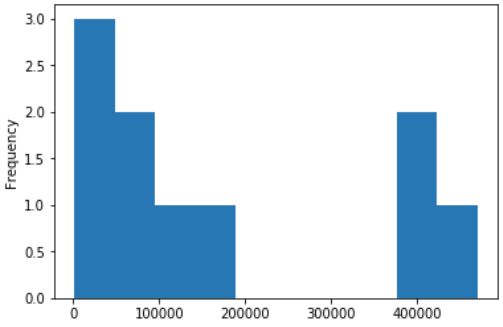
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 11 columns):
Fiscal
Year                  10 non-null int64
Type                          10 non-null object
Request
Accepted            10 non-null int64
Request
Rejected            10 non-null int64
Total Request Received    10 non-null int64
Average Accepted/Day      10 non-null int64
Biometrics Scheduled      2 non-null float64
Request Under Review      2 non-null float64
Approved                  10 non-null int64
Denied                    10 non-null object
Pending                   10 non-null int64
dtypes: float64(2), int64(7), object(2)
memory usage: 960.0+ bytes
```

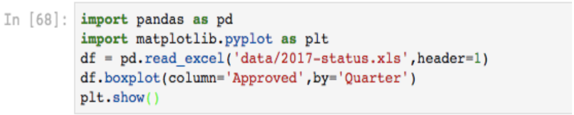
DETECT OUTLIERS USING DATA VISUALIZATION

Using histogram checked the number of approved case status for all the cases and the number does looks fine.

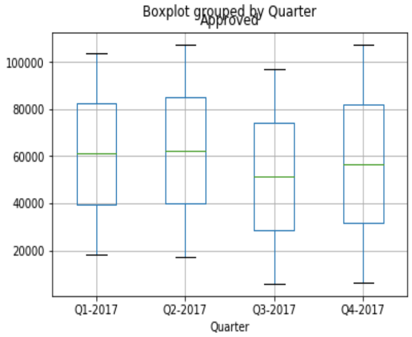


```
import pandas as pd
df = pd.read_excel('data/case-status.xls',header=1)
df.Approved.plot('hist')
import matplotlib.pyplot as plt
plt.show()
```





```
In [68]: import pandas as pd
         import matplotlib.pyplot as plt
         df = pd.read_excel('data/2017-status.xls',header=1)
         df.boxplot(column='Approved',by='Quarter')
         plt.show()
```

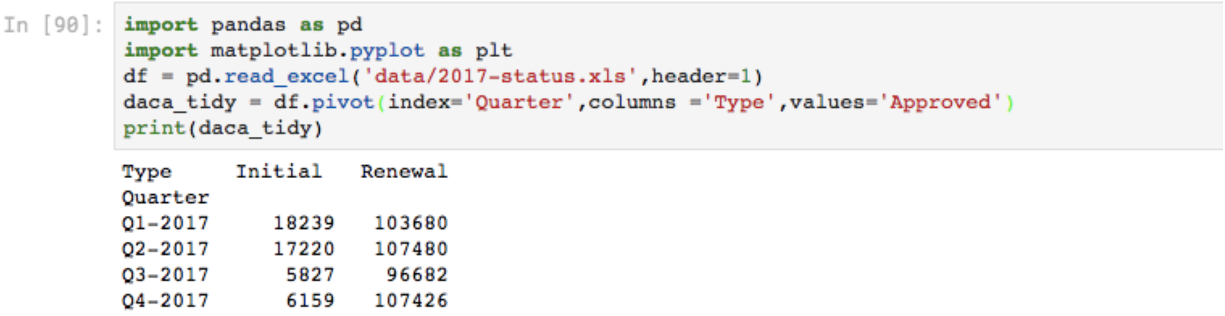


Tidy Data

- Columns represent separate variables
- Rows represent individual observations
- Observation units from table

PIVOT: UN-MELTING DATA

The raw data provided by the USCIS website was not normalized. So, I had to normalize the data while converting it into excel. This was done prior to importing the dataset in python. But I had to pivot the data to group by year to convert the data from Analysis friendly shape to reporting friendly shape.



```
In [90]: import pandas as pd
         import matplotlib.pyplot as plt
         df = pd.read_excel('data/2017-status.xls',header=1)
         daca_tidy = df.pivot(index='Quarter',columns ='Type',values='Approved')
         print(daca_tidy)

         Type      Initial   Renewal
         Quarter
         Q1-2017     18239    103680
         Q2-2017     17220    107480
         Q3-2017      5827     96682
         Q4-2017      6159    107426
```

PIVOT TABLE METHOD

This method was not needed as there were no duplicates in any of the datasets.