Introduction

This notebook contains the data preprocessing steps for the semester project on People's Analytics. The goal of this project is to analyse real data from fellow selection process at Global Health Corps to recommend strategies to optimize their application review process.

Data Preprocessing

In [467]:
```
# Import all the libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

Let's load the data set for applicant's info and take a quick look at it.

In [138]:
```
# Load the dataset
data = pd.read_excel('Selection.xlsx',sheet_name='Application Info')
```

In [468]:
```
data.head()
```

Out[468]:

	UniqueAppID	Position	Birth Year	Birth Month	Birth Day	Citizenship Status	Other Citizenship Status	Sex	Other Sex	W in
0	A184	R05-Int	1991	7	13	United States	NaN	Female	NaN	Ye mo tha ye
1	A346	R09-Int	1991	11	22	United States	NaN	Female	NaN	Ye ye
2	A714	G09-Int	1989	11	24	United States	NaN	Prefer not to respond	NaN	Ye ye
3	A914	G09-Int	1989	6	4	United States	NaN	Female	NaN	No ha ne he or int in
4	A1003	R09-Int	1992	11	10	United States	NaN	Female	NaN	Ye ye

Quite a few number of columns....Let's check out the dimensions of the dataset.

In [470]:
```
print data.shape
```

```
(5778, 76)
```

The dataset contains 5778 rows with 76 columns. Let's check out the datatypes and if null values are present in the dataset...

In [471]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5778 entries, 0 to 5777
Data columns (total 76 columns):
UniqueAppID
5778 non-null object
Position
5778 non-null object
Birth Year
5778 non-null int64
Birth Month
5778 non-null int64
Birth Day
5778 non-null int64
Citizenship Status
5778 non-null object
Other Citizenship Status
1139 non-null object
Sex
5778 non-null object
Other Sex
2 non-null object
Worked in public health
5778 non-null object
Studied public health
5778 non-null object
American Indian
35 non-null object
```

```
Asian
659 non-null object
Black
2965 non-null object
Hispanic
278 non-null object
Hawaiian
25 non-null object
White
1502 non-null object
Other Race
533 non-null object
Have you previously applied?
5778 non-null object
Language 1
5778 non-null object
Language other 1
163 non-null object
Language proficiency 1
5778 non-null object
Language 2
5383 non-null object
Language other 2
813 non-null object
Language proficiency 2
5383 non-null object
Language 3
3512 non-null object
Language other 3
1177 non-null object
Language proficiency 3
3512 non-null object
Round2 Reviews Plagirism Reviewer1
5613 non-null object
Round2 Clear Purpose CF Reviewer1
5613 non-null float64
Round2 Commitment to Social Justice Reviewer1
5613 non-null float64
Round2 Innovation CF Reviewer1
```

```
5613 non-null float64
Round2 Commitment to learning2 Reviewer1
5613 non-null float64
Round2 Get Results Reviewer1
5613 non-null float64
Round2 Collaboration Reviewer1
5613 non-null object
Round2 Inspire and Mobilize Reviewer1
5613 non-null object
Round2 Experience Reviewer1
5613 non-null float64
Round2 Total Score Reviewer1
5613 non-null float64
Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer1
5613 non-null object
Round2 GHC Semifinalist Reviewer1
5613 non-null float64
Round2 Is this applicant moving on as a GHC Alternate? Reviewer1
5613 non-null object
Round2 GHC Alternate Reviewer1
5613 non-null float64
Round2 R2ReviewerID Reviewer1
5613 non-null object
Round2 Plagirism Reviewer2
5613 non-null object
Round2 Clear Purpose CF Reviewer2
5613 non-null float64
Round2 Commitment to Social Justice CF  Reviewer2
5613 non-null float64
Round2 Innovation CF Reviewer2
5613 non-null float64
Round2 Commitment to learning2 Reviewer2
5613 non-null float64
Round2 Get Results Reviewer2
5613 non-null float64
Round2 Collaboration Reviewer2
5613 non-null object
Round2 Inspire and Mobilize Reviewer2
5613 non-null object
```

```
Round2 Experience Reviewer2
5613 non-null float64
Round2 Total Score Reviewer2
5613 non-null float64
Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer2
5613 non-null object
Round2 GHC Semifinalist Reviewer2
5613 non-null float64
Round2 Is this applicant moving on as a GHC Alternate? Reviewer2
5613 non-null object
Round2 GHC Alternate Reviewer2
5613 non-null float64
Round2 R2ReviewerID2 Reviewer2
4250 non-null object
Round3 Semifinalist Designation
5614 non-null object
Round3 UniqueStaffID
1376 non-null object
Round4 Clear Purpose
1224 non-null object
Round4 Commitment to Social Justice
1224 non-null float64
Round4 Collboration
1224 non-null float64
Round4 Inspire and Mobilize
1224 non-null float64
Round4 Adapt and Innovate
1224 non-null float64
Round4 Commitment to Learning
1224 non-null float64
Round4 Get Results
1224 non-null float64
Round4 Cross-Cultural Awareness/Sensitivity
1224 non-null float64
Round4 Experience/Transferable Skills
1224 non-null float64
Round4 Total
1224 non-null float64
Round4 GHC Finalist? Only 3-5 applicants per positions
```

```
1224 non-null object
Round4 Finalist Ranking
1224 non-null float64
Round4 Other Commitments
1224 non-null object
Round4 ReviewerID
939 non-null object
Round5 Partner Ranking
610 non-null object
Round5 PartnerOrgID
610 non-null object
dtypes: float64(28), int64(3), object(45)
memory usage: 3.4+ MB
```

The above information table shows quite a lot of null values and categorical columns. Let's handle the individual columns one by one by creating dummy variables for categorical columns and removing the missing values.

In [172]:
```
# Make a  copy of the data set
df = data
```

Citizenship status of the applicants is an important aspect in the selection process and this variable is categorical in nature. Let's start with creating dummy variable for this column.

In [173]:
```
# Citizenship status : dummy variable creation
Native=df['Citizenship Status'].str.get_dummies(sep=',')
Native.columns = ['Citizen' + '_' + str(col) for col in Native.columns]
# merge in the original dataset
df=df.merge(Native, left_index=True, right_index=True)
# drop the original columns
df.drop(['Citizenship Status'],axis=1,inplace=True)
```

Let's verify the dummy variables created above and merged in the original dataset....

In [472]:
```
# Dummy variables for citizenship status
```

```
Native.head()
```

Out[472]:

	Citizen_Other	Citizen_Rwanda	Citizen_Uganda	Citizen_United States	Citizen_Zambia	Citizen_Malawi	Citizen_Other	Citizen_R
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0

Whether the applicant worked in public health sector is categorical in nature with four unique values. These values are long strings. Let's map these values to a more interpretable and short string values using dictionary mapping and then create dummy variables for them.

In [174]:
```
# Worked in public health column
df['Worked in public health'].unique()
```

Out[174]:
```
array([u'Yes, more than 3 years', u'Yes, 1-3 years',
       u'No, I have never held a job or internship in public health',
       u'Yes, less than 1 year'], dtype=object)
```

In [175]:
```
# Worked in public health : mapping the values to a shorter string valu
es
publicHealth = {'Yes, more than 3 years':'greater than 3 years',
                'Yes, 1-3 years':'Between 1 to 3 years',
                'No, I have never held a job or internship in public he
alth':'Never',
                'Yes, less than 1 year':'less than 1 year'}
df['PublicHealth'] = df['Worked in public health'].map(publicHealth)

#Dummy variable creation
```

```
pb=df['PublicHealth'].str.get_dummies()
pb.columns = ['WorkedPublicHealth' + '_' + str(col).replace(' ', '_') f
or col in pb.columns]

# merge the dummy variables in the original dataset and drop the irrele
vant columns
df = df.merge(pb,left_index=True, right_index=True)
df.drop(['Worked in public health','PublicHealth'],axis=1,inplace=True)
```

Likewise , whether an applicant studied public health is a categorical column with three unique values. Let's handle it in the similar fashion.

In [178]:
```
# Studied public health
df['Studied public health'].unique()
```

Out[178]:
```
array([u'No, I have never studied public health',
       u'Yes, it was my major/minor/focus area',
       u'Yes, I took at least one public health class'], dtype=object)
```

In [179]:
```
# Studied in public health : mapping the values to shorter string value
s

StpublicHealth = {'No, I have never studied public health':'Never',
                  'Yes, it was my major/minor/focus area': 'More than 1
 Class',
                  'Yes, I took at least one public health class': 'Atlea
st 1 Class' }

# dictionary mapping of shorter string values
df['StPublicHealth'] = df['Studied public health'].map(StpublicHealth)

# Dummy variable creation
spb = df['StPublicHealth'].str.get_dummies()
spb.columns = ['StudiedPublicHealth' + '_' + str(col).replace(' ', '_')
 for col in spb.columns]

# merge the dummy variables in the original dataset and drop the irrele
vant columns
```

```
df = df.merge(spb,left_index=True, right_index=True)
df.drop(['Studied public health','StPublicHealth'],axis=1,inplace=True)
```

Next there are seven different columns for the ethnicity of the applicants and they all contain missing values. Let's fill the missing values with boolean value of zero and one hot encode them.

In [182]:
```
# Function to handle race columns
def handleRace(colnames,df):
    for col in colnames:
        df[col].fillna(0,inplace=True)
        value=df[col].unique()[-1]
        df[col].replace(str(value),1,inplace=True)
    return df

# Different race column names
colnames = ['American Indian','Asian','Black', 'Hispanic','Hawaiian','O
ther Race']
df=handleRace(colnames,df)
```

Let's verify the above operation....

In [185]:
```
df['American Indian'].unique()
```

Out[185]:
```
array([0, 1])
```

In [186]:
```
df['Asian'].unique()
```

Out[186]:
```
array([0, 1])
```

In [187]:
```
df['White'].unique()
```

Out[187]:
```
array([u'White', nan], dtype=object)
```

White column was not handled above. So, let's handle it separately below...

In [188]:
```
# Handling white racial ethnicity column
```

```
df['White'].fillna(0,inplace=True)
df['White'].replace('White',1,inplace=True)
```

In [191]:
```
# Have you ever applied
df['Have you previously applied?'].unique()
```

Out[191]:
```
array([u'No', u'Yes'], dtype=object)
```

In [197]:
```
# map the string values to boolean values
app = {'No':'0',
       'Yes':'1'}

df['Have you previously applied?']=df['Have you previously applied?'].m
ap(app)
```

It makes sense to combine the first language column and other language column of the applicants. Let's go ahead and do that.

In [201]:
```
# no. of Unique languages as first language before the merge operation.
len(df['Language 1'].unique())
```

Out[201]:
```
11
```

In [208]:
```
# combine the language 1 and other language 1 into first language colum
n
df['firstLanguage']= np.where(df['Language 1'] == 'Other', df['Language
 other 1'], df['Language 1'])
```

In [210]:
```
# No. of unique first languages.
df['firstLanguage'].nunique()
```

Out[210]:
```
49
```

Let's create the dummy variables now...

In [220]:
```
# first language : dummy variable creation
```

```
fl = df['firstLanguage'].str.get_dummies()
fl.columns = ['firstLanguage' + '_' + col for col in fl.columns]
# merge the dummy variables in the original dataset and drop the irrele
vant columns
df = df.merge(fl,left_index=True, right_index=True)
df.drop(['firstLanguage','Language 1','Language other 1'],axis=1,inplac
e=True)
```

In [231]:
```
# Language proficiency : dummy variable creation
prof=df['Language proficiency 1'].str.get_dummies()
df = df.merge(prof,left_index=True, right_index=True)
df.drop(['Language proficiency 1'],axis=1,inplace=True)
```

There are round 5 different rounds in the fellow selection process. In this journal entry we are focusing on round 1,round 2 and round 3 to classify candidates if they qualify for semifinalist round. In the above code section of the notebook, I cleaned the basic columns related to the applicants information.

Moving further, we are going to clean up and handle Round 2 based columns which contain values/scores that were given by the two reviewers involved in round 2 to the applicants....

Let's start with reviewer1...

Round 2 Reviewer 1

Round 2 based columns contain missing values. Moreover, couple columns contain string values like "Not enough information". The whole idea here is to fill the missing values, replace "not enough information" values with boolean value of zero where ever required and convert the string values of "yes" and "no" into boolean values....

In [233]:
```
# Round2 Reviews Plagirism Reviewer1
df['Round2 Reviews Plagirism Reviewer1'].unique()
```

Out[233]:
```
array([u'No', 0, nan, u'Yes'], dtype=object)
```

In [241]:
```
# Round2 Reviews Plagirism Reviewer1 : fill missing values and map the
 string values into boolean values
df['Round2 Reviews Plagirism Reviewer1'].fillna(0,inplace=True)
df['Round2 Reviews Plagirism Reviewer1'] = df['Round2 Reviews Plagirism
 Reviewer1'].map(app)
```

In [244]:
```
# Round2 Clear Purpose CF Reviewer1
df['Round2 Clear Purpose CF Reviewer1'].unique()
```

Out[244]:
```
array([ 4.,  3.,  2.,  1.,  0., nan])
```

In [245]:
```
# Round2 Clear Purpose CF Reviewer1 : fill missing values
df['Round2 Clear Purpose CF Reviewer1'].fillna(0,inplace=True)
```

In [246]:
```
# Round2 Commitment to Social Justice Reviewer1
df['Round2 Commitment to Social Justice Reviewer1'].unique()
```

Out[246]:
```
array([ 4.,  3.,  2.,  1.,  0., nan])
```

In [248]:
```
# Round2 Commitment to Social Justice Reviewer1 : fill missing values
df['Round2 Commitment to Social Justice Reviewer1'].fillna(0,inplace=Tr
ue)
```

In [249]:
```
# Round2 Innovation CF Reviewer1
df['Round2 Innovation CF Reviewer1'].unique()
```

Out[249]:
```
array([ 3.,  4.,  2.,  1.,  0., nan])
```

In [251]:
```
# Round2 Innovation CF Reviewer1 : fill missing values
df['Round2 Innovation CF Reviewer1'].fillna(0,inplace=True)
```

In [237]:
```
# Round2 Commitment to learning2 Reviewer1
df['Round2 Commitment to learning2 Reviewer1'].unique()
```

Out[237]:
```
array([ 3.,  4.,  2.,  1.,  0., nan])
```

In [253]:
```
# Round2 Commitment to learning2 Reviewer1 : fill missing values
```

```
df['Round2 Commitment to learning2 Reviewer1'].fillna(0,inplace=True)
```

In [238]:
```
# Round2 Get Results Reviewer1
df['Round2 Get Results Reviewer1'].unique()
```

Out[238]:
```
array([ 3.,  4.,  2.,  1.,  0., nan])
```

In [255]:
```
# Round2 Get Results Reviewer1 : fill missing values
df['Round2 Get Results Reviewer1'].fillna(0,inplace=True)
```

In [239]:
```
# Round2 Collaboration Reviewer1
df['Round2 Collaboration Reviewer1'].unique()
```

Out[239]:
```
array([3, 4, u'Not enough information', 2, 1, 0, nan], dtype=object)
```

In [256]:
```
# Round2 Collaboration Reviewer1 : fill missing values and replace not
 enough information with 0
df['Round2 Collaboration Reviewer1'].fillna(0,inplace=True)
df['Round2 Collaboration Reviewer1'] = np.where(df['Round2 Collaboratio
n Reviewer1'] == 'Not enough information', 0, df['Round2 Collaboration
 Reviewer1'])
```

In [258]:
```
# Round2 Inspire and Mobilize Reviewer1
df['Round2 Inspire and Mobilize Reviewer1'].unique()
```

Out[258]:
```
array([3, 4, 2, u'Not enough information', 1, 0, nan], dtype=object)
```

In [259]:
```
# Round2 Inspire and Mobilize Reviewer1 : fill missing values and repla
ce not enough information with 0
df['Round2 Inspire and Mobilize Reviewer1'].fillna(0,inplace=True)
df['Round2 Inspire and Mobilize Reviewer1'] = np.where(df['Round2 Inspi
re and Mobilize Reviewer1'] == 'Not enough information', 0, df['Round2
 Inspire and Mobilize Reviewer1'])
```

In [261]:
```
# Round2 Experience Reviewer1
df['Round2 Experience Reviewer1'].unique()
```

Out[261]:
```
array([ 4.,  3.,  2.,  1.,  0., nan])
```

In [262]:
```
# Round2 Experience Reviewer1 : fill missing values
df['Round2 Experience Reviewer1'].fillna(0,inplace=True)
```

In [265]:
```
# Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer1
df['Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer
1'].unique()
```

Out[265]:
```
array([u'Yes', u'No', 0, nan], dtype=object)
```

In [267]:
```
#Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer1 :
 filling missing values and map string values of yes and no into boolea
n
df['Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer
1'].fillna(0,inplace=True)
df['Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer
1'] = df['Round2 Is this applicant moving on as a GHC Semi-Finalist Rev
iewer1'].map(app)
```

In [271]:
```
# Round2 Is this applicant moving on as a GHC Alternate? Reviewer1
df['Round2 Is this applicant moving on as a GHC Alternate? Reviewer1'].
unique()
```

Out[271]:
```
array([u'No', u'Yes', 0, nan], dtype=object)
```

In [272]:
```
# Round2 Is this applicant moving on as a GHC Alternate? Reviewer1 : fi
lling missing values and map string values of yes and no
df['Round2 Is this applicant moving on as a GHC Alternate? Reviewer1'].
fillna(0,inplace=True)
df['Round2 Is this applicant moving on as a GHC Alternate? Reviewer1']
= df['Round2 Is this applicant moving on as a GHC Alternate? Reviewer1'
].map(app)
```

In [274]:
```
# Round2 GHC Alternate Reviewer1
df['Round2 GHC Alternate Reviewer1'].unique()
```

Out[274]:
```
array([ 0.,  1.,  2.,  3.,  4.,  5., nan])
```

In [275]:
```
# Round2 GHC Alternate Reviewer1 : fill missing values
df['Round2 GHC Alternate Reviewer1'].fillna(0,inplace=True)
```

Round 2 Reviewer 2

Like wise let's handle the round 2 columns for reviewer 2.

In [276]:
```
# Round2 Plagirism Reviewer2
df['Round2 Plagirism Reviewer2'].unique()
```

Out[276]:
```
array([0, u'No', u'Yes', nan], dtype=object)
```

In [428]:
```
# Round2 Plagirism Reviewer2 : fill missing values and map string value
s into boolean
df['Round2 Plagirism Reviewer2'].fillna(0,inplace=True)
df['Round2 Plagirism Reviewer2'] = df['Round2 Plagirism Reviewer2'].map
(app)
```

In [282]:
```
# Round2 Clear Purpose CF Reviewer2
df['Round2 Clear Purpose CF Reviewer2'].unique()
```

Out[282]:
```
array([ 0.,  3.,  4.,  2.,  1., nan])
```

In [283]:
```
# Round2 Clear Purpose CF Reviewer2 : fill missing values
df['Round2 Clear Purpose CF Reviewer2'].fillna(0,inplace=True)
```

In [284]:
```
# Round2 Commitment to Social Justice CF  Reviewer2
df['Round2 Commitment to Social Justice CF  Reviewer2'].unique()
```

Out[284]:
```
array([ 0.,  3.,  4.,  2.,  1., nan])
```

In [285]:
```
# Round2 Commitment to Social Justice CF  Reviewer2 : fill missing valu
es
df['Round2 Commitment to Social Justice CF  Reviewer2'].fillna(0,inplac
e=True)
```

In [286]:
```
# Round2 Innovation CF Reviewer2
df['Round2 Innovation CF Reviewer2'].unique()
```

Out[286]:
```
array([ 0.,  2.,  3.,  4.,  1., nan])
```

In [287]:
```
# Round2 Innovation CF Reviewer2 : fill the missing values
df['Round2 Innovation CF Reviewer2'].fillna(0,inplace=True)
```

In [288]:
```
# Round2 Commitment to learning2 Reviewer2
df['Round2 Commitment to learning2 Reviewer2'].unique()
```

Out[288]:
```
array([ 0.,  3.,  4.,  2.,  1., nan])
```

In [289]:
```
# Round2 Commitment to learning2 Reviewer2 : fill missing values
df['Round2 Commitment to learning2 Reviewer2'].fillna(0,inplace=True)
```

In [290]:
```
# Round2 Get Results Reviewer2
df['Round2 Get Results Reviewer2'].unique()
```

Out[290]:
```
array([ 0.,  3.,  4.,  2.,  1., nan])
```

In [291]:
```
# Round2 Get Results Reviewer2 : fill missing values
df['Round2 Get Results Reviewer2'].fillna(0,inplace=True)
```

In [292]:
```
# Round2 Collaboration Reviewer2
df['Round2 Collaboration Reviewer2'].unique()
```

Out[292]:
```
array([0, u'Not enough information', 4, 3, 2, 1, nan], dtype=object)
```

In [293]:
```
# Round2 Collaboration Reviewer2 : fill missing values and replace not
 enough information with 0
df['Round2 Collaboration Reviewer2'].fillna(0,inplace=True)
df['Round2 Collaboration Reviewer2'] = np.where(df['Round2 Collaboratio
n Reviewer2'] == 'Not enough information', 0, df['Round2 Collaboration
 Reviewer2'])
```

In [294]:
```
# Round2 Inspire and Mobilize Reviewer2
df['Round2 Inspire and Mobilize Reviewer2'].unique()
```

Out[294]:
```
array([0, u'Not enough information', 4, 3, 2, 1, nan], dtype=object)
```

In [295]:
```
# Round2 Inspire and Mobilize Reviewer2 : fill missing values and repla
ce not enough information with 0
df['Round2 Inspire and Mobilize Reviewer2'].fillna(0,inplace=True)
df['Round2 Inspire and Mobilize Reviewer2'] = np.where(df['Round2 Inspi
re and Mobilize Reviewer2'] == 'Not enough information', 0, df['Round2
 Inspire and Mobilize Reviewer2'])
```

In [296]:
```
# Round2 Experience Reviewer2
df['Round2 Experience Reviewer2'].unique()
```

Out[296]:
```
array([ 0.,  3.,  4.,  1.,  2., nan])
```

In [297]:
```
#  Round2 Experience Reviewer2 : fill missing values
df['Round2 Experience Reviewer2'].fillna(0,inplace=True)
```

In [299]:
```
# Round2 Total Score Reviewer2
df['Round2 Total Score Reviewer2'].unique()
```

Out[299]:
```
array([ 0., 17., 27., 25., 20., 31., 30., 26., 24., 29., 28., 22., 15.,
       21., 18., 19., 14., 32., 23., 16., 12.,  6., 13., 11., 10.,  7.,
        9., nan,  8.])
```

In [300]:
```
# Round2 Total Score Reviewer2 : filling missing values
df['Round2 Total Score Reviewer2'].fillna(0,inplace=True)
```

In [301]:
```
# Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer2
df['Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer
2'].unique()
```

Out[301]:
```
array([0, u'Yes', u'No', nan], dtype=object)
```

In [302]:
```
# Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer2 :
```

```
 fill missing values and map string values into boolean values
df['Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer
2'].fillna(0,inplace=True)
df['Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer
2'] = df['Round2 Is this applicant moving on as a GHC Semi-Finalist Rev
iewer2'].map(app)
```

In [303]:
```
# Round2 GHC Semifinalist Reviewer2
df['Round2 GHC Semifinalist Reviewer2'].unique()
```

Out[303]:
```
array([ 0., 10.,  3.,  4.,  5.,  9.,  2.,  1.,  8.,  7.,  6., nan])
```

In [304]:
```
# Round2 GHC Semifinalist Reviewer2 : fill missing values
df['Round2 GHC Semifinalist Reviewer2'].fillna(0,inplace=True)
```

In [305]:
```
# Round2 Is this applicant moving on as a GHC Alternate? Reviewer2
df['Round2 Is this applicant moving on as a GHC Alternate? Reviewer2'].
unique()
```

Out[305]:
```
array([0, u'No', u'Yes', nan], dtype=object)
```

In [306]:
```
# Round2 Is this applicant moving on as a GHC Alternate? Reviewer2: fil
l missing values
df['Round2 Is this applicant moving on as a GHC Alternate? Reviewer2'].
fillna(0,inplace=True)
```

In [307]:
```
# Round2 GHC Alternate Reviewer2
df['Round2 GHC Alternate Reviewer2'].unique()
```

Out[307]:
```
array([ 0.,  1.,  3.,  2.,  4.,  5., nan])
```

In [308]:
```
# Round2 GHC Alternate Reviewer2 : fill missing values
df['Round2 GHC Alternate Reviewer2'].fillna(0,inplace=True)
```

The above couple sections were the scores given by round 2 reviewers based on candidate's profile.The next column is "Round3 Semifinalist Designation" which is the final decision made by round 3 reviewer about whether the applicant/candidate is going to the semifinalist round or not.

Essentially, round 3 reviewer reconciles any discrepancy between the two reviewers of round 2, evaluates the candidates and provides his decision..

In [311]:
```
# Round3 Semifinalist Designation
df['Round3 Semifinalist Designation'].value_counts()
```

Out[311]:
```
0            4238
Yes          1283
Alternate      47
No             46
Name: Round3 Semifinalist Designation, dtype: int64
```

Round 3 semifinalist designation column contains 'alternate" values as well. We are only interested in finding out whether the applicant enters semifinals or not. Moreover, there are only 47 rows with 'alternate' values. So, let's remove such rows and prepare the dataset for binary classification problem where **Round 3 semifinalist designation** column will serve as the target variable.

In [317]:
```
# Removing rows with alternate values and creating a new dataframe
dfnew=df.loc[np.where(df['Round3 Semifinalist Designation'] != 'Alterna
te')]
```

In [320]:
```
# Map the string values in target variable into boolean values
dfnew['Round3 Semifinalist Designation'] = dfnew['Round3 Semifinalist D
esignation'].map(app)
dfnew['Round3 Semifinalist Designation'].fillna(0,inplace=True)
dfnew['Round3 Semifinalist Designation'].replace(0,'0',inplace=True)
```

In [329]:
```
dfnew['Round3 Semifinalist Designation'].value_counts()
```

Out[329]:
```
0    4448
1    1283
Name: Round3 Semifinalist Designation, dtype: int64
```

Let's look at other remaining columns like Gender of the candidates....

In [384]:
```
dfnew['Sex'].value_counts()
```

Out[384]:
```
Female                    3763
Male                      1847
Prefer not to respond       18
Other                        3
Name: Sex, dtype: int64
```

In [387]:
```
# Removing rows with Prefer not ro respond and Other values
dfnew=dfnew.iloc[np.where(dfnew['Sex'] != 'Prefer not to respond')]
dfnew = dfnew.iloc[np.where(dfnew['Sex'] != 'Other')]
```

There are missing values as well. Let's assume they are female candidates. It would be interesting to see if Gender of the candidates impacts their selection process.

In [378]:
```
dfnew['Sex'].fillna('Female',inplace=True)
```

Finally let's go ahead and compile all the columns that will be used as features for model training....

In [405]:
```
# features compilation from applicant info
raceCols = dfnew.columns[8:15]
basicCols = dfnew.columns[[6,15]]
round2Cols = dfnew.columns[22:52]
targetCol = dfnew.columns[52]
citizenshipCols = dfnew.columns[70:81]
publicHealthCols = dfnew.columns[81:88]
langCols = dfnew.columns[88:]
```

In [424]:
```
# Prepare a new dataset with relevant feature columns
final=dfnew[raceCols]
final=final.merge(dfnew[basicCols],left_index=True,right_index=True)
final=final.merge(dfnew[round2Cols],left_index=True,right_index=True)
final=final.merge(dfnew[citizenshipCols],left_index=True,right_index=Tr
ue)
final=final.merge(dfnew[publicHealthCols],left_index=True,right_index=T
```

```
rue)
final=final.merge(dfnew[langCols],left_index=True,right_index=True)
```

In [462]:
```
# Include the target variable
final['SemifinalistDesignation'] = dfnew[targetCol]
```

Let's a look at how the final processed data looks like....

In [425]:
```
final
```

Out[425]:

	American Indian	Asian	Black	Hispanic	Hawaiian	White	Other Race	Sex	Have you previously applied?	R R Pla Rev
0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	0	0
1	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	1	0
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Female	NaN	NaN
3	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	0	0
4	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	1	0
5	0.0	0.0	0.0	1.0	0.0	1.0	0.0	Female	0	0
6	0.0	1.0	0.0	0.0	0.0	0.0	0.0	Female	0	0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Female	0	0
8	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	0	0

9	0.0	0.0	0.0	1.0	0.0	1.0	0.0	Female	0	0
10	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	0	0
11	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Female	0	0
12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Female	0	0
13	0.0	1.0	0.0	0.0	0.0	0.0	0.0	Female	0	0
14	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	0	0
15	0.0	1.0	0.0	0.0	0.0	0.0	0.0	Female	0	0
16	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Male	0	0
17	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	0	0
18	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	0	0
19	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	1	0
20	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Female	0	0
21	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Female	0	0
22	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	1	0
23	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	1	0
24	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Female	0	0
25	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	0	0
26	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	1	0
27	0.0	0.0	0.0	1.0	0.0	0.0	0.0	Male	0	0
28	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Male	1	0
29	0.0	0.0	1.0	0.0	0.0	0.0	1.0	Male	1	0
...	...	...	...	...	...	...	...	...	...	...
5625	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Male	1	0

5626	0.0	0.0	1.0	0.0	0.0	0.0	1.0	Male	0	0
5627	0.0	1.0	0.0	0.0	0.0	0.0	0.0	Female	0	0
5628	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Male	0	0
5629	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	0	0
5630	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Female	0	0
5631	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Female	0	0
5632	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	0	0
5633	0.0	0.0	0.0	0.0	0.0	0.0	1.0	Female	0	0
5634	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Female	0	0
5635	0.0	1.0	0.0	0.0	0.0	1.0	0.0	Female	0	0
5636	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Female	1	0
5637	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	0	0
5638	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	0	0
5639	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Female	0	0
5640	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Male	0	0
5641	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Female	0	NaN
5642	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Male	1	0
5643	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Male	0	0
5644	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Female	1	0
5645	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	1	0
5646	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Female	0	0
5647	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Male	0	0
5648	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Male	1	0

5649	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Female	0	0
5650	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	0	0
5651	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Female	0	0
5653	0.0	0.0	1.0	0.0	0.0	0.0	0.0	Male	0	0
5654	0.0	0.0	0.0	1.0	0.0	0.0	0.0	Female	0	0
5655	0.0	0.0	0.0	0.0	0.0	1.0	0.0	Female	0	0

5618 rows × 111 columns

Still there are missing values....Let's check more!!

In [426]:
```
# check missing values
def check_nan(list_values):
    return sum(list_values.isnull())

#Using apply function to check missing values on every column of datafr
ame
print final.apply(check_nan)
```

```
American Indian
  83
Asian
  83
Black
  83
Hispanic
  83
Hawaiian
  83
White
  83
Other Race
  83
```

```
Sex
   8
Have you previously applied?
  83
Round2 Reviews Plagirism Reviewer1
494
Round2 Clear Purpose CF Reviewer1
  83
Round2 Commitment to Social Justice Reviewer1
  83
Round2 Innovation CF Reviewer1
  83
Round2 Commitment to learning2 Reviewer1
  83
Round2 Get Results Reviewer1
  83
Round2 Collaboration Reviewer1
  83
Round2 Inspire and Mobilize Reviewer1
  83
Round2 Experience Reviewer1
  83
Round2 Total Score Reviewer1
245
Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer1
494
Round2 GHC Semifinalist Reviewer1
245
Round2 Is this applicant moving on as a GHC Alternate? Reviewer1
494
Round2 GHC Alternate Reviewer1
  83
Round2 R2ReviewerID Reviewer1
245
Round2 Plagirism Reviewer2
1561
Round2 Clear Purpose CF Reviewer2
  83
Round2 Commitment to Social Justice CF  Reviewer2
```

```
  83
Round2 Innovation CF Reviewer2
  83
Round2 Commitment to learning2 Reviewer2
  83
Round2 Get Results Reviewer2
  83

...
firstLanguage_Kirundi
  83
firstLanguage_Korean
  83
firstLanguage_Lithuanian
  83
firstLanguage_Luganda
  83
firstLanguage_Luo
  83
firstLanguage_Mongolian
  83
firstLanguage_Nepalese
  83
firstLanguage_Nepali
  83
firstLanguage_Nyanja
  83
firstLanguage_Portuguese
  83
firstLanguage_Rukiga
  83
firstLanguage_Runyankole
  83
firstLanguage_Runyankore
  83
firstLanguage_Russian
  83
firstLanguage_Serbian language
  83
```

```
firstLanguage_Shona
  83
firstLanguage_Somali
  83
firstLanguage_Spanish
  83
firstLanguage_Swahili
  83
firstLanguage_Swedish
  83
firstLanguage_Sylheti
  83
firstLanguage_Tigrigna
  83
firstLanguage_Tonga
  83
firstLanguage_Urdu
  83
firstLanguage_Vietnamese
  83
Advanced/Full Professional Proficiency
  83
Basic Proficiency
  83
Fluent/Native Speaker
  83
Limited working proficiency
  83
Professional working proficiency
  83
Length: 111, dtype: int64
```

It makes sense to handle this mess right here....

In [431]:
```
final['Round2 Plagirism Reviewer2'].fillna(0,inplace=True)
```

In [432]:
```
final['Round2 Plagirism Reviewer2'].unique()
```

Out[432]:
```
array([0, '0', '1'], dtype=object)
```

In [433]:
```
final['Round2 Plagirism Reviewer2'].replace(0,'0',inplace=True)
```

In [434]:
```
final['Round2 Plagirism Reviewer2'].unique()
```

Out[434]:
```
array(['0', '1'], dtype=object)
```

In [435]:
```
final['Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer1'].unique()
```

Out[435]:
```
array(['1', '0', nan], dtype=object)
```

In [453]:
```
final['Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer1'].fillna('0',inplace=True)
final['Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer2'].fillna('0',inplace=True)
```

In [437]:
```
final['Round2 Is this applicant moving on as a GHC Alternate? Reviewer1'].unique()
```

Out[437]:
```
array(['0', nan, '1'], dtype=object)
```

In [438]:
```
final['Round2 Is this applicant moving on as a GHC Alternate? Reviewer1'].fillna('0',inplace=True)
```

In [ ]:
```
toRemove = ['Round2 R2ReviewerID Reviewer1','Round2 R2ReviewerID Reviewer2']
```

In [441]:
```
del final['Round2 R2ReviewerID Reviewer1']
del final['Round2 R2ReviewerID2 Reviewer2']
```

In [442]:
```
print final.apply(check_nan)
```

```
American Indian
83
```

```
Asian
83
Black
83
Hispanic
83
Hawaiian
83
White
83
Other Race
83
Sex
  8
Have you previously applied?
83
Round2 Reviews Plagirism Reviewer1
494
Round2 Clear Purpose CF Reviewer1
83
Round2 Commitment to Social Justice Reviewer1
83
Round2 Innovation CF Reviewer1
83
Round2 Commitment to learning2 Reviewer1
83
Round2 Get Results Reviewer1
83
Round2 Collaboration Reviewer1
83
Round2 Inspire and Mobilize Reviewer1
83
Round2 Experience Reviewer1
83
Round2 Total Score Reviewer1
245
Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer1
  0
Round2 GHC Semifinalist Reviewer1
```

```
245
Round2 Is this applicant moving on as a GHC Alternate? Reviewer1
  0
Round2 GHC Alternate Reviewer1
83
Round2 Plagirism Reviewer2
  0
Round2 Clear Purpose CF Reviewer2
83
Round2 Commitment to Social Justice CF  Reviewer2
83
Round2 Innovation CF Reviewer2
83
Round2 Commitment to learning2 Reviewer2
83
Round2 Get Results Reviewer2
83
Round2 Collaboration Reviewer2
83

...
firstLanguage_Kirundi
83
firstLanguage_Korean
83
firstLanguage_Lithuanian
83
firstLanguage_Luganda
83
firstLanguage_Luo
83
firstLanguage_Mongolian
83
firstLanguage_Nepalese
83
firstLanguage_Nepali
83
firstLanguage_Nyanja
83
```

```
firstLanguage_Portuguese
83
firstLanguage_Rukiga
83
firstLanguage_Runyankole
83
firstLanguage_Runyankore
83
firstLanguage_Russian
83
firstLanguage_Serbian language
83
firstLanguage_Shona
83
firstLanguage_Somali
83
firstLanguage_Spanish
83
firstLanguage_Swahili
83
firstLanguage_Swedish
83
firstLanguage_Sylheti
83
firstLanguage_Tigrigna
83
firstLanguage_Tonga
83
firstLanguage_Urdu
83
firstLanguage_Vietnamese
83
Advanced/Full Professional Proficiency
83
Basic Proficiency
83
Fluent/Native Speaker
83
Limited working proficiency
```

```
83
Professional working proficiency
83
Length: 109, dtype: int64
```

In [443]:
```
final['Round2 Reviews Plagirism Reviewer1'].unique()
```

Out[443]:
```
array(['0', nan, '1'], dtype=object)
```

In [444]:
```
final['Round2 Reviews Plagirism Reviewer1'].fillna('0',inplace=True)
```

In [446]:
```
final['Round2 Total Score Reviewer1'].fillna(0,inplace=True)
```

In [448]:
```
final['Round2 GHC Semifinalist Reviewer1'].unique()
```

Out[448]:
```
array([ 6.,  0., nan,  2., 10.,  3.,  4.,  7.,  9.,  5.,  1.,  8.])
```

In [449]:
```
final['Round2 GHC Semifinalist Reviewer1'].fillna(0,inplace=True)
```

In [464]:
```
final.apply(check_nan)[0:50]
```

Out[464]:
```
American Indian
83
Asian
83
Black
83
Hispanic
83
Hawaiian
83
White
83
Other Race
83
Sex
8
Have you previously applied?
```

```
83
Round2 Reviews Plagirism Reviewer1
0
Round2 Clear Purpose CF Reviewer1
83
Round2 Commitment to Social Justice Reviewer1
83
Round2 Innovation CF Reviewer1
83
Round2 Commitment to learning2 Reviewer1
83
Round2 Get Results Reviewer1
83
Round2 Collaboration Reviewer1
83
Round2 Inspire and Mobilize Reviewer1
83
Round2 Experience Reviewer1
83
Round2 Total Score Reviewer1
0
Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer1
0
Round2 GHC Semifinalist Reviewer1
0
Round2 Is this applicant moving on as a GHC Alternate? Reviewer1
0
Round2 GHC Alternate Reviewer1
83
Round2 Plagirism Reviewer2
0
Round2 Clear Purpose CF Reviewer2
83
Round2 Commitment to Social Justice CF  Reviewer2
83
Round2 Innovation CF Reviewer2
83
Round2 Commitment to learning2 Reviewer2
83
```

```
Round2 Get Results Reviewer2
83
Round2 Collaboration Reviewer2
83
Round2 Inspire and Mobilize Reviewer2
83
Round2 Experience Reviewer2
83
Round2 Total Score Reviewer2
83
Round2 Is this applicant moving on as a GHC Semi-Finalist Reviewer2
0
Round2 GHC Semifinalist Reviewer2
83
Round2 Is this applicant moving on as a GHC Alternate? Reviewer2
83
Round2 GHC Alternate Reviewer2
83
Citizen_ Other
83
Citizen_ Rwanda
83
Citizen_ Uganda
83
Citizen_ United States
83
Citizen_ Zambia
83
Citizen_Malawi
83
Citizen_Other
83
Citizen_Rwanda
83
Citizen_Uganda
83
Citizen_United States
83
Citizen_Zambia
```

```
83
WorkedPublicHealth_Between_1_to_3_years
83
WorkedPublicHealth_Never
83
dtype: int64
```

In [465]:
```
final.apply(check_nan)[50:]
```

Out[465]:
```
WorkedPublicHealth_greater_than_3_years    83
WorkedPublicHealth_less_than_1_year        83
StudiedPublicHealth_Atleast_1_Class        83
StudiedPublicHealth_More_than_1_Class      83
StudiedPublicHealth_Never                  83
firstLanguage_Amharic                      83
firstLanguage_Arabic                       83
firstLanguage_Arabic (Juba)                83
firstLanguage_Bahasa Melayu                83
firstLanguage_Bangla                       83
firstLanguage_Belarusian                   83
firstLanguage_Bemba                        83
firstLanguage_Bengali                      83
firstLanguage_Chichewa                     83
firstLanguage_Chinese                      83
firstLanguage_Chitonga                     83
firstLanguage_Danish                       83
firstLanguage_Dutch                        83
firstLanguage_English                      83
firstLanguage_English and Gujarati         83
firstLanguage_EÊ‹egbe                      83
firstLanguage_Farsi                        83
firstLanguage_French                       83
firstLanguage_German                       83
firstLanguage_Hindi                        83
firstLanguage_Hungarian                    83
firstLanguage_Italian                      83
firstLanguage_KINYARWANDA                  83
firstLanguage_Kinyarwanda                  83
firstLanguage_Kirundi                      83
```

```
firstLanguage_Korean                          83
firstLanguage_Lithuanian                      83
firstLanguage_Luganda                         83
firstLanguage_Luo                             83
firstLanguage_Mongolian                       83
firstLanguage_Nepalese                        83
firstLanguage_Nepali                          83
firstLanguage_Nyanja                          83
firstLanguage_Portuguese                      83
firstLanguage_Rukiga                          83
firstLanguage_Runyankole                      83
firstLanguage_Runyankore                      83
firstLanguage_Russian                         83
firstLanguage_Serbian language                83
firstLanguage_Shona                           83
firstLanguage_Somali                          83
firstLanguage_Spanish                         83
firstLanguage_Swahili                         83
firstLanguage_Swedish                         83
firstLanguage_Sylheti                         83
firstLanguage_Tigrigna                        83
firstLanguage_Tonga                           83
firstLanguage_Urdu                            83
firstLanguage_Vietnamese                      83
Advanced/Full Professional Proficiency        83
Basic Proficiency                             83
Fluent/Native Speaker                         83
Limited working proficiency                   83
Professional working proficiency              83
SemifinalistDesignation                       83
dtype: int64
```

Let's save this dataset and use it for machine learning in the next notebook..

In [466]:
```
# Saving the preprocessed data into a csv file....
final.to_csv('final.csv', sep='\t', encoding='utf-8')
```

Discussions :

This notebook contains the data preprocessing operation for People's Analytics project. Only a part of feature variables have been processed that are primarily about the basic profile of the applicants, round 2 reviews and round 3 review. In this journal entry, we are focusing only on implementing tree based ensemble methods to classify candidates whether they are entering semifinal round or not.

The machine learning part of this journal entry can be found in "Machine Learning" Notebook.