Data Processing: ACS Demographics

Group 42: Manish Vuyyuru, Victor Sheng, Elise Penn, Yajaira Gonzalez

Processes the ACs demographics. Needs access to the files available on the team GitHub folder, which are too large and numerous to upload here.

In [1]:
```
#imports
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import seaborn as sns
import numpy as np
import pandas as pd
import pickle
%matplotlib inline
```

In [2]:
```
# dictionary to translate numeric codes to state abbrev. names
state_codes = {
1: "AL", 2: "AK",4: "AZ",5: "AR",6: "CA",8: "CO",9: "CT",10: "DE",
11: "DC",12: "FL",13: "GA",15: "HI",16: "ID",17: "IL",18: "IN",19: "IA",
20: "KS",21: "KY",22: "LA",23: "ME",24: "MD",25: "MA",26: "MI",27: "MN",
28: "MS",29: "MO",30: "MT",31: "NE",32: "NV",33: "NH",34: "NJ",35: "NM",
36: "NY",37: "NC",38: "ND",39: "OH",40: "OK",41: "OR",42: "PA",44: "RI",
45: "SC",46: "SD",47: "TN",48: "TX",49: "UT",50: "VT",51: "VA",53: "WA",
54: "WV",55: "WI",56: "WY"}
```

In [3]:
```
# raw ACS data file paths
path10 = 'Datasets/demographics/ACS_10_1YR_S0201_with_ann.csv'
path12 = 'Datasets/demographics/ACS_12_1YR_S0201_with_ann.csv'
path14 = 'Datasets/demographics/ACS_14_1YR_S0201_with_ann.csv'
path16 = 'Datasets/demographics/ACS_16_1YR_S0201_with_ann.csv'
path17 = 'Datasets/demographics/ACS_17_1YR_S0201_with_ann.csv'

# read in the raw files
df10 = pd.read_csv(path10, header = 1)
df12 = pd.read_csv(path12, header = 1)
df14 = pd.read_csv(path14, header = 1)
df16 = pd.read_csv(path16, header = 1)
df17 = pd.read_csv(path17, header = 1)
```

In [4]:
```
def clean(df, year):

    # insert year column
    df.insert(0, 'year', year)

    # insert state column (as 2-char code, like: MA)
    df.insert(1, 'state', df.Id2.apply(lambda x: int(x / 100)))
    df.state = df.state.apply(lambda x: state_codes[x])

    # insert district column, as an integer
    # districts should start with district 1, not 0
    df.insert(2, 'district', df.Id2.apply(lambda x: int(str(x)[-2:])))
    df.district = df.district.replace(0, 1)

    # filter out Margin of Error columns
    columns_to_keep = [col for col in df.columns if not 'Margin of Error;' in col
    df = df[columns_to_keep]

    # create index as: state_district_year
    df.index = ['{0}_{1:02d}_{2}'.format(row['state'], row['district'], row['year

    return df
```

In [5]:
```
acs10 = clean(df10, 2010)
acs12 = clean(df12, 2012)
acs14 = clean(df14, 2014)
acs16 = clean(df16, 2016)
acs17 = clean(df17, 2017)
```

In [6]:
```
# check order of unemployement columns
[col for col in acs10.columns if 'EMPLOYMENT STATUS - In labor force - Civilian l
```

Out[6]:
```
['Estimate; EMPLOYMENT STATUS - In labor force - Civilian labor force - Unemployed
',
 'Estimate; EMPLOYMENT STATUS - In labor force - Civilian labor force - Unemployed
- Percent of civilian labor force',
 'Estimate; EMPLOYMENT STATUS - In labor force - Civilian labor force - Unemployed
.1',
 'Estimate; EMPLOYMENT STATUS - In labor force - Civilian labor force - Unemployed
- Percent of civilian labor force.1']
```

In [7]:
```
len(acs17.columns)
```

Out[7]:
```
316
```

In [8]:
```
# find the common set of columns between two dataframes
len(set(acs16.columns).intersection(set(acs17.columns)))
```

Out[8]:
```
287
```

In [9]:
```
1 set(acs10.columns).intersection(set(acs12.columns))
```

Out[9]:
```
{'Estimate; CLASS OF WORKER - Civilian employed population 16 years and over',
 'Estimate; CLASS OF WORKER - Government workers',
 'Estimate; CLASS OF WORKER - Private wage and salary workers',
 'Estimate; CLASS OF WORKER - Self-employed workers in own not incorporated busine
ss',
 'Estimate; CLASS OF WORKER - Unpaid family workers',
 'Estimate; COMMUTING TO WORK - Car, truck, or van - carpooled',
 'Estimate; COMMUTING TO WORK - Car, truck, or van - drove alone',
 'Estimate; COMMUTING TO WORK - Mean travel time to work (minutes)',
 'Estimate; COMMUTING TO WORK - Other means',
 'Estimate; COMMUTING TO WORK - Public transportation (excluding taxicab)',
 'Estimate; COMMUTING TO WORK - Walked',
 'Estimate; COMMUTING TO WORK - Worked at home',
 'Estimate; COMMUTING TO WORK - Workers 16 years and over',
 'Estimate; DISABILITY STATUS - Civilian noninstitutionalized population 18 to 64
years',
 'Estimate; DISABILITY STATUS - Civilian noninstitutionalized population 65 years
and older',
 'Estimate; DISABILITY STATUS - Civilian noninstitutionalized population under 18
```

In [10]:
```
1 #[col for col in df.columns if not 'Margin of Error;' in col]
2
```

In [11]:

```
# 2010 and 2012
raw_predictor_name_subset_10_12 = [
"Estimate; SEX AND AGE - Female",
"Estimate; SEX AND AGE - 18 to 24 years",
"Estimate; SEX AND AGE - 25 to 34 years",
"Estimate; SEX AND AGE - Median age (years)",
"Estimate; RELATIONSHIP - Nonrelatives - Unmarried partner",
"Estimate; HOUSEHOLDS BY TYPE - Nonfamily households - Male householder - Living
"Estimate; EDUCATIONAL ATTAINMENT - Bachelor's degree or higher",
"Estimate; FERTILITY - Women 15 to 50 years who had a birth in the past 12 months
"Estimate; VETERAN STATUS - Civilian veteran",
"Estimate; RESIDENCE 1 YEAR AGO - Same house",
"Estimate; PLACE OF BIRTH, CITIZENSHIP STATUS AND YEAR OF ENTRY - Native",
"Estimate; PLACE OF BIRTH, CITIZENSHIP STATUS AND YEAR OF ENTRY - Foreign born",
"Estimate; WORLD REGION OF BIRTH OF FOREIGN BORN - Latin America",
"Estimate; LANGUAGE SPOKEN AT HOME AND ABILITY TO SPEAK ENGLISH - Language other
"Estimate; EMPLOYMENT STATUS - In labor force - Civilian labor force - Unemployed
"Estimate; COMMUTING TO WORK - Public transportation (excluding taxicab)",
"Estimate; HEALTH INSURANCE COVERAGE - No health insurance coverage",
"Estimate; POVERTY RATES FOR FAMILIES AND PEOPLE FOR WHOM POVERTY STATUS IS DETER
"Estimate; OWNER CHARACTERISTICS - Median value (dollars)"]

raw_predictor_name_supplement_10 = [
"Estimate; INCOME IN THE PAST 12 MONTHS (IN 2010 INFLATION-ADJUSTED DOLLARS) - Me
"Estimate; INCOME IN THE PAST 12 MONTHS (IN 2010 INFLATION-ADJUSTED DOLLARS) - Wi

raw_predictor_name_supplement_12 = [
"Estimate; INCOME IN THE PAST 12 MONTHS (IN 2012 INFLATION-ADJUSTED DOLLARS) - Me
"Estimate; INCOME IN THE PAST 12 MONTHS (IN 2012 INFLATION-ADJUSTED DOLLARS) - Wi

raw_predictor_names_10 = raw_predictor_name_subset_10_12 + raw_predictor_name_sup
raw_predictor_names_12 = raw_predictor_name_subset_10_12 + raw_predictor_name_sup


# 2014 and beyond
raw_predictor_name_subset_14_beyond = [
"Estimate; SEX AND AGE - Total population - Female",
"Estimate; SEX AND AGE - 18 to 24 years",
"Estimate; SEX AND AGE - 25 to 34 years",
"Estimate; SEX AND AGE - Median age (years)",
"Estimate; RELATIONSHIP - Population in households - Nonrelatives - Unmarried par
"Estimate; HOUSEHOLDS BY TYPE - Households - Nonfamily households - Male househol
"Estimate; EDUCATIONAL ATTAINMENT - Bachelor's degree or higher",
"Estimate; FERTILITY - Women 15 to 50 years - Women 15 to 50 years who had a birt
"Estimate; VETERAN STATUS - Civilian population 18 years and over - Civilian vete
"Estimate; RESIDENCE 1 YEAR AGO - Population 1 year and over - Same house",
"Estimate; PLACE OF BIRTH, CITIZENSHIP STATUS AND YEAR OF ENTRY - Native",
"Estimate; PLACE OF BIRTH, CITIZENSHIP STATUS AND YEAR OF ENTRY - Foreign born",
"Estimate; WORLD REGION OF BIRTH OF FOREIGN BORN - Foreign-born population exclud
"Estimate; LANGUAGE SPOKEN AT HOME AND ABILITY TO SPEAK ENGLISH - Population 5 ye
"Estimate; EMPLOYMENT STATUS - Population 16 years and over - In labor force - Ci
"Estimate; COMMUTING TO WORK - Workers 16 years and over - Public transportation
"Estimate; HEALTH INSURANCE COVERAGE - Civilian noninstitutionalized population -
"Estimate; POVERTY RATES FOR FAMILIES AND PEOPLE FOR WHOM POVERTY STATUS IS DETER
"Estimate; OWNER CHARACTERISTICS - Owner-occupied housing units - Median value (d

raw_predictor_name_supplement_14 = [
"Estimate; INCOME IN THE PAST 12 MONTHS (IN 2014 INFLATION-ADJUSTED DOLLARS) - Ho
"Estimate; INCOME IN THE PAST 12 MONTHS (IN 2014 INFLATION-ADJUSTED DOLLARS) - Wi

raw_predictor_name_supplement_16 = [
"Estimate; INCOME IN THE PAST 12 MONTHS (IN 2016 INFLATION-ADJUSTED DOLLARS) - Ho
"Estimate; INCOME IN THE PAST 12 MONTHS (IN 2016 INFLATION-ADJUSTED DOLLARS) - Wi
```

In [12]:
```
renamed_predictors = [
"female_pct",
"age18_24_pct",
"age25_34_pct",
"median_age",
"unmarried_partner_pct",
"male_living_alone_pct",
"bachelors_deg_or_higher_pct",
"past_year_births_to_unmarried_women_pct",
"civilian_veteran_pct",
"live_same_house_past_year_pct",
"native_born_population",
"foreign_born_population",
"foreign_born_proportion_from_LatinAmerica",
"speak_other_language_at_home_pct",
"labor_force_unemployed_pct",
"public_transit_commuter_pct",
"no_health_insurance_pct",
"poverty_rate_pct",
"median_housing_value",
"median_household_income",
"food_stamp_benefits_pct"]
```

In [13]:
```
# check number of predictors
len(acs17[raw_predictor_names_17].columns)
```

Out[13]: 21

In [14]:
```
identifier_cols = ['state', 'district', 'year']

acs10 = acs10[['state', 'district', 'year'] + raw_predictor_names_10]
acs12 = acs12[['state', 'district', 'year'] + raw_predictor_names_12]
acs14 = acs14[['state', 'district', 'year'] + raw_predictor_names_14]
acs16 = acs16[['state', 'district', 'year'] + raw_predictor_names_16]
acs17 = acs17[['state', 'district', 'year'] + raw_predictor_names_17]
```

In [15]:
```
acs10.columns = identifier_cols + renamed_predictors
acs10.head()
```

Out[15]:

	state	district	year	female_pct	age18_24_pct	age25_34_pct	median_age	unmarried_partner_pct	n
AL_01_2010	AL	1	2010	51.0	9.0	12.3	38.4	1.6	
AL_02_2010	AL	2	2010	52.1	10.1	12.6	37.7	1.7	
AL_03_2010	AL	3	2010	51.7	12.5	12.1	36.7	1.6	
AL_04_2010	AL	4	2010	50.6	8.4	11.3	40.2	1.2	
AL_05_2010	AL	5	2010	51.0	9.8	12.4	38.3	1.2	

5 rows × 24 columns

In [16]:
```
# convert foreign / native born population to ratio
for df in [acs10, acs12, acs14, acs16, acs17]:
    df.columns = identifier_cols + renamed_predictors
    df['foreign_to_native_born_ratio'] = df['foreign_born_population'] / df['nati
    df.drop(['native_born_population', 'foreign_born_population'], axis = 1, inpl
```

In [17]:
```
# use 2017 demographics as 2018 data
acs18 = acs17
acs18.year = 2018
acs18.index = ['{0}_{1:02d}_{2}'.format(row['state'], row['district'], row['year'
```

In [18]:
```
acs18
```

Out[18]:

	state	district	year	female_pct	age18_24_pct	age25_34_pct	median_age	unmarried_partner_pct	ma
AL_01_2018	AL	1	2018	51.8	8.2	12.9	40.0	1.4	
AL_02_2018	AL	2	2018	51.2	9.7	13.0	38.5	1.7	
AL_03_2018	AL	3	2018	51.2	10.7	13.0	38.1	2.1	
AL_04_2018	AL	4	2018	51.4	8.8	11.7	40.7	1.3	
AL_05_2018	AL	5	2018	50.8	9.3	13.2	39.5	1.6	
AL_06_2018	AL	6	2018	52.3	8.6	12.3	39.1	1.3	
AL_07_2018	AL	7	2018	52.3	12.5	14.2	36.3	1.6	
AK_01_2018	AK	1	2018	47.9	9.9	16.0	34.5	2.7	
AZ_01_2018	AZ	1	2018	49.5	10.7	12.2	37.6	2.3	
AZ_02_2018	AZ	2	2018	50.5	9.5	12.8	41.4	3.0	
AZ_03_2018	AZ	3	2018	50.0	13.1	14.1	31.8	2.6	

In [19]:
```
# stack 5 dataframes from each year on top of each other
demographics_data = acs18.append([acs16, acs14, acs12, acs10])
```

In [20]:
```
demographics_data
```

Out[20]:

	state	district	year	female_pct	age18_24_pct	age25_34_pct	median_age	unmarried_partner_pct	ma
AL_01_2018	AL	1	2018	51.8	8.2	12.9	40.0	1.4	
AL_02_2018	AL	2	2018	51.2	9.7	13.0	38.5	1.7	
AL_03_2018	AL	3	2018	51.2	10.7	13.0	38.1	2.1	
AL_04_2018	AL	4	2018	51.4	8.8	11.7	40.7	1.3	
AL_05_2018	AL	5	2018	50.8	9.3	13.2	39.5	1.6	
AL_06_2018	AL	6	2018	52.3	8.6	12.3	39.1	1.3	
AL_07_2018	AL	7	2018	52.3	12.5	14.2	36.3	1.6	
AK_01_2018	AK	1	2018	47.9	9.9	16.0	34.5	2.7	
AZ_01_2018	AZ	1	2018	49.5	10.7	12.2	37.6	2.3	
AZ_02_2018	AZ	2	2018	50.5	9.5	12.8	41.4	3.0	
AZ_03_2018	AZ	3	2018	50.0	13.1	14.1	31.8	2.6	

In [21]:
```
demographics_data.to_csv('Datasets/demographics_data_2010_to_2018.csv')
```

In [22]:
```
1 demographics_data.columns
```

Out[22]:
```
Index(['state', 'district', 'year', 'female_pct', 'age18_24_pct',
       'age25_34_pct', 'median_age', 'unmarried_partner_pct',
       'male_living_alone_pct', 'bachelors_deg_or_higher_pct',
       'past_year_births_to_unmarried_women_pct', 'civilian_veteran_pct',
       'live_same_house_past_year_pct',
       'foreign_born_proportion_from_LatinAmerica',
       'speak_other_language_at_home_pct', 'labor_force_unemployed_pct',
       'public_transit_commuter_pct', 'no_health_insurance_pct',
       'poverty_rate_pct', 'median_housing_value', 'median_household_income',
       'food_stamp_benefits_pct', 'foreign_to_native_born_ratio'],
      dtype='object')
```

In [23]:
```
1 fec = pickle.load(open('Datasets/data_FEC_NATIONALPOLL_2004_2018.p', 'rb'))
```

In [24]:
```
1 fec.index
```

Out[24]:
```
Index(['AK_01_2004', 'AL_01_2004', 'AL_02_2004', 'AL_03_2004', 'AL_04_2004',
       'AL_05_2004', 'AL_06_2004', 'AL_07_2004', 'AR_01_2004', 'AR_02_2004',
       ...
       'WV_03_2018', 'WI_01_2018', 'WI_02_2018', 'WI_03_2018', 'WI_04_2018',
       'WI_05_2018', 'WI_06_2018', 'WI_07_2018', 'WI_08_2018', 'WY_01_2018'],
      dtype='object', length=3431)
```

In [25]:
```
1 demographics_data = demographics_data.drop(['district', 'state', 'year'], axis=1)
```

In [26]:
```
1 # try joining FEC election data with ACS demographic data
2
3
4 fec_demographics_data = fec.join(demographics_data, how='inner')
```

In [27]:
```
1 set(fec_demographics_data['year'])
```

Out[27]:
```
{2010, 2012, 2014, 2016, 2018}
```

In [28]:
```
1 # pickle.dump(fec_demographics_data, open('Datasets/data_FEC_NATIONALPOLL_DEMOGRA
2 # fec_demographics_data.to_csv('Datasets/data_FEC_NATIONALPOLL_DEMOGRAPHICS_2010_
3 # test = pickle.load(open('Datasets/data_FEC_NATIONALPOLL_DEMOGRAPHICS_2010_2018.
4 # fec_demographics_data.shape, test.shape
```

In [29]:
```
1 fec.shape, demographics_data.shape
```

Out[29]:
```
((3431, 22), (2180, 20))
```

In [30]:
```
1 fec_demographics_imputed_data = fec.join(demographics_data, how='outer')
2
3 for index, row in fec_demographics_imputed_data.iterrows():
4     if index not in fec.index:
5         fec_demographics_imputed_data = fec_demographics_imputed_data[fec_demogra
6
7 fec_demographics_imputed_data.shape
```

Out[30]:
```
(3431, 42)
```

In [31]:
```
1 assert(sorted(fec_demographics_imputed_data.index) == sorted(fec.index))
```

In [32]:
```
demographics_data.columns
```

Out[32]:
```
Index(['female_pct', 'age18_24_pct', 'age25_34_pct', 'median_age',
       'unmarried_partner_pct', 'male_living_alone_pct',
       'bachelors_deg_or_higher_pct',
       'past_year_births_to_unmarried_women_pct', 'civilian_veteran_pct',
       'live_same_house_past_year_pct',
       'foreign_born_proportion_from_LatinAmerica',
       'speak_other_language_at_home_pct', 'labor_force_unemployed_pct',
       'public_transit_commuter_pct', 'no_health_insurance_pct',
       'poverty_rate_pct', 'median_housing_value', 'median_household_income',
       'food_stamp_benefits_pct', 'foreign_to_native_born_ratio'],
      dtype='object')
```

In [33]:
```
for index, row in fec_demographics_imputed_data.iterrows():
    if row['year'] < 2010:
        impute_index = index[:-5] + '_2010'
        fec_demographics_imputed_data.loc[fec_demographics_imputed_data.index ==
```

In [34]:
```
fec_demographics_imputed_data[fec_demographics_imputed_data.index == 'AK_01_2010'
```

Out[34]:

	district	state	year	party	candidatevotes	totalvotes	candidate	national_poll	national_poll_p
AK_01_2010	1.0	AK	2010.0	republican	175384	254335	Don Young	-2.622642	9.824

1 rows × 42 columns

In [35]:
```
fec_demographics_imputed_data[fec_demographics_imputed_data.index == 'AK_01_2004'
```

Out[35]:

	district	state	year	party	candidatevotes	totalvotes	candidate	national_poll	national_poll_p
AK_01_2004	1.0	AK	2004.0	republican	213216	299996	Don Young	3.680556	-0.989

1 rows × 42 columns

In [36]:
```
for col in demographics_data.columns:
    if fec_demographics_imputed_data[col].isnull().values.any():
        raise ValueError
```

In [37]:
```
set(fec_demographics_imputed_data['year'].values)
```

Out[37]:
```
{2004.0, 2006.0, 2008.0, 2010.0, 2012.0, 2014.0, 2016.0, 2018.0}
```

In [38]:
```
pickle.dump(fec_demographics_imputed_data, open('Datasets/data_FEC_NATIONALPOLL_I
fec_demographics_imputed_data.to_csv('Datasets/data_FEC_NATIONALPOLL_DEMOGRAPHICS
test = pickle.load(open('Datasets/data_FEC_NATIONALPOLL_DEMOGRAPHICSIMPUTED_2004_
fec_demographics_imputed_data.shape, test.shape
```

Out[38]:
```
((3431, 42), (3431, 42))
```

In [39]:
```
set(test['year'].values)
```

Out[39]:
```
{2004.0, 2006.0, 2008.0, 2010.0, 2012.0, 2014.0, 2016.0, 2018.0}
```

In [40]:
```
for col in demographics_data.columns:
    if test[col].isnull().values.any():
        raise ValueError
```

In [ ]:
```
1 fec[fec.index == 'WI_06_2018']
```

In [ ]:
```
1 demographics_data[demographics_data.index == 'WI_06_2018']
```

In [ ]:
```
1 fec_demographics_data[fec_demographics_data.index == 'WI_06_2018']
```

In [ ]:
```
1 test.columns
```

In [ ]:
```
1 fec.shape
```

In [ ]:
```
1 demographics_data.shape
```

In [ ]:
```
1 demographics_data.columns
```

In [ ]:
```
1
```