Data Processing: House Preprocessing

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

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

In [ ]:
```python
import pandas as pd
import numpy as np
from functions import houseFunctions as hfunc
import pickle

pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)
```

In [2]:
```python
houseResFile = "Datasets/fec/1976-2016-house.csv"
```

In [3]:
```python
winners_df, winners2_df  = hfunc.load_data(houseResFile, minYear=2002)
winners_df, winners2_df = hfunc.clean_index(winners_df), hfunc.clean_index(winner
#data = hfunc.fetch_trimmed_data(winners_df, winners2_df, minYear=2004)
```

In [4]:
```python
#data.head()
```

In [5]:
```python
#set(data['year'].values)
```

In [6]:
```python
houseResFile = "Datasets/fec/2018wiki-12072018.csv"
wiki2018 = pd.read_csv(houseResFile, header=None)
wiki2018.columns = ['location', 'PVI', 'representative', 'party', 'first_elected'
wiki2018 = wiki2018[['location', 'representative', 'results', 'candidates']]
wiki2018['location'] = wiki2018['location'].str.strip()

wiki2018_clean = pd.DataFrame()
wiki2018_tmp = wiki2018.copy()
wiki2018_tmp['location'] = wiki2018_tmp['location'].fillna(method='ffill', inplac
for key, shard in wiki2018_tmp.groupby(['location']):
    shard = shard.dropna(axis=0, subset=['candidates'])
    shard = shard[shard['candidates'].str.contains('√')]
    wiki2018_clean = wiki2018_clean.append(shard)
wiki2018 = wiki2018_clean
wiki2018_clean.head()
```

Out[6]:

	location	representative	results	candidates
0	Alabama 1	Bradley Byrne	Incumbent re-elected.	√ Bradley Byrne (Republican) 63.3%[64]
2	Alabama 2	Martha Roby	Incumbent re-elected.	√ Martha Roby (Republican) 61.5%[64]
4	Alabama 3	Mike Rogers	Incumbent re-elected.	√ Mike Rogers (Republican) 63.8%[64]
6	Alabama 4	Robert Aderholt	Incumbent re-elected.	√ Robert Aderholt (Republican) 79.9%[64]
8	Alabama 5	Mo Brooks	Incumbent re-elected.	√ Mo Brooks (Republican) 61.1%[64]

In [7]:
```
wiki2018[wiki2018['location'].str.contains('Washington')]
```

Out[7]:

	location	representative	results	candidates
1124	Washington 1	NaN	NaN	√ Suzan DelBene (Democratic)[208]
1143	Washington 10	NaN	NaN	√ Denny Heck (Democratic)[208]
1125	Washington 2	Rick Larsen	Incumbent re-elected.	√ Rick Larsen (Democratic)[208]
1127	Washington 3	Jaime Herrera Beutler	Incumbent re-elected.	√ Jaime Herrera Beutler (Republican)[208]
1130	Washington 4	NaN	NaN	√ Dan Newhouse (Republican)[208]
1132	Washington 5	NaN	NaN	√ Cathy McMorris Rodgers (Republican)[208]
1134	Washington 6	NaN	NaN	√ Derek Kilmer (Democratic)[208]
1135	Washington 7	Pramila Jayapal	Incumbent re-elected.	√ Pramila Jayapal (Democratic) 83.4%[208]
1138	Washington 8	NaN	New member elected.	√ Kim Schrier (Democratic)[208][209]
1140	Washington 9	Adam Smith	Incumbent re-elected.	√ Adam Smith (Democratic)[208]

In [8]:
```
state_names = np.array(['ALABAMA', 'ALASKA', 'ARIZONA', 'ARKANSAS',
'CALIFORNIA',
                        'COLORADO', 'CONNECTICUT', 'DELAWARE', 'FLORIDA', 'GEORGIA',
                        'HAWAII', 'IDAHO', 'ILLINOIS', 'INDIANA', 'IOWA', 'KANSAS',
                        'KENTUCKY', 'LOUISIANA', 'MAINE', 'MARYLAND', 'MASSACHUSETTS',
                        'MICHIGAN', 'MINNESOTA', 'MISSISSIPPI', 'MISSOURI', 'MONTANA',
                        'NEBRASKA', 'NEVADA', 'NEW HAMPSHIRE', 'NEW JERSEY',
'NEW MEXICO',
                        'NEW YORK', 'NORTH CAROLINA', 'NORTH DAKOTA', 'OHIO',
'OKLAHOMA',
                        'OREGON', 'PENNSYLVANIA', 'RHODE ISLAND', 'SOUTH CAROLINA',
                        'SOUTH DAKOTA', 'TENNESSEE', 'TEXAS', 'UTAH', 'VERMONT',
                        'VIRGINIA', 'WASHINGTON', 'WEST VIRGINIA', 'WISCONSIN',
'WYOMING'])

state_abbrs = np.array(['AL','AK','AZ','AR','CA','CO','CT','DE','FL','GA','HI','I
                       'IN','IA','KS','KY','LA','ME','MD','MA','MI','MN','MS','MO','MT',
                       'NE','NV','NH','NJ','NM','NY','NC','ND','OH','OK','OR','PA','RI',
                       'SC','SD','TN','TX','UT','VT','VA','WA','WV','WI','WY'])
```

In [9]:
```
def clean_location(row):
    index_0_string = 'at-large'
    if row['location'][-len(index_0_string):] == index_0_string:
        row['district'] = 1
        row['state'] = row['location'][:-len(index_0_string)].strip()
    else:
        index = None
        for cursor, char in enumerate(row['location'][::-1]):
            if not char.isnumeric():
                index = cursor
        row['district'] = row['location'][index-1:].strip()
        row['state'] = row['location'][:index-1].strip()

    row['state'] = state_abbrs[np.where(state_names == row['state'].upper())][0]
    row['year'] = 2018
    row['party'] = row['candidates'][row['candidates'].find('(')+1:row['candidate

    row['candidatevotes'] = None
    row['totalvotes'] = None
    row['candidate'] = None

    return row
wiki2018 = wiki2018.apply(clean_location, axis=1).drop('location', axis=1)
wiki2018.columns, wiki2018.shape
```

Out[9]:
```
(Index(['representative', 'results', 'candidates', 'district', 'state', 'year', 'p
arty', 'candidatevotes', 'totalvotes', 'candidate'], dtype='object'),
 (434, 10))
```

In [19]:
```
wiki2018['candidate'].isnull().values.any()
```

Out[19]:
```
True
```

In [ ]:
```
wiki2018 = hfunc.clean_index(wiki2018, clean_before_build=False)
```

In [ ]:
```
hfunc.fetch_index(winners_df, wiki2018, save=True, load=False)
```

In [ ]:
```
winners_df.columns
```

In [ ]:
```
wiki2018.columns, wiki2018.dtypes
```

In [ ]:
```
wiki2018['party'] = wiki2018['party'].str.lower()
wiki2018.loc[wiki2018['party'] == 'democratic', 'party'] = 'democrat'
```

In [ ]:
```
common_cols = ['candidate', 'candidatevotes', 'district', 'party', 'state', 'tota
winners_df = pd.concat([winners_df, wiki2018[common_cols]])
winners2_df = pd.concat([winners2_df, wiki2018[common_cols]])

data = hfunc.fetch_trimmed_data(winners_df, winners2_df, minYear=2004)
```

In [ ]:
```
winners_df.dtypes
```

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

In [ ]:
```
data[(data.isnull().any(axis=1)) & (~data['year'] == 2018)]
```

In [ ]:
```
pickle.dump(data, open('Datasets/data_FEC_NATIONALPOLL_2004_2018.p', 'wb'))
data.to_csv('Datasets/data_FEC_NATIONALPOLL_2004_2018.csv')
```

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

In [ ]:
```
import pandas as pd
import numpy as np
from functions import houseFunctions as hfunc
import pickle
dataset = pickle.load(open('Datasets/data.p', 'rb'))
```

In [ ]:
```
dataset = dataset.loc[:,['dem_win', 'dem_win_prev', 'margin_signed_minus_prev', '
dataset.columns
```

In [ ]:
```
# %reset
```

In [ ]:
```
subset2018 = dataset[dataset['year'] == 2018]
np.sum(subset2018['dem_win'] != subset2018['dem_win_prev']), np.sum(subset2018['c
```

In [ ]:
```

```