

Workforce Diversity Public Firms

November 18, 2025

```
[1]: import pandas as pd
import swifter
pd.set_option('display.max_columns', 500)
pd.set_option('display.max_rows', 500)
```

1 Read files

```
[2]: df_position_US = pd.read_pickle("Processed_Data/df_position_US_clean.pkl")
```

```
[3]: df_position_US.head()
```

```
[3]:
```

	user_id	position_id	company_cleaned	\
25	user_id_147639296	position_id_-4598965781019268056	citrix	
26	user_id_147639296	position_id_-3538691097286546776	citrix	
29	user_id_91670979	position_id_3263106306747497228	citrix	
30	user_id_91670979	position_id_6131126283879609552	citrix	
31	user_id_91670979	position_id_5123644626084198358	citrix	

	region	country	state	msa	startdate	\
25	Northern America	United States	NC	Raleigh-Cary NC MSA	2020-02-01	
26	Northern America	United States	NC	Raleigh-Cary NC MSA	2021-03-01	
29	Northern America	United States	NC	Raleigh-Cary NC MSA	2015-09-01	
30	Northern America	United States	NC	Raleigh-Cary NC MSA	2018-01-01	
31	Northern America	United States	NC	Raleigh-Cary NC MSA	2021-03-01	

	enddate	jobtitle_raw	salary	\
25	2021-03-01	Sales Development Representative	54894.16	
26	2021-11-01	Inside Territory Manager- Greenfield Enterprise	60591.276	
29	NaN	Webinar Programs\, Marketing	62851.027	
30	NaN	Adoption Marketing Manager	77010.485	
31	2022-05-01	Senior Adoption Marketing Manager	93054.006	

	seniority	rn	rcid	onet_code	cusip	gvkey	len_startdate	\
25	1.0	3.0	rcid_509865	NaN	177376100	61676	10	
26	2.0	4.0	rcid_509865	NaN	177376100	61676	10	
29	2.0	4.0	rcid_509865	NaN	177376100	61676	10	
30	4.0	6.0	rcid_509865	NaN	177376100	61676	10	

31	4.0	7.0	rcid_509865	NaN	177376100	61676	10
----	-----	-----	-------------	-----	-----------	-------	----

	d_date	enddate2
25	1	2021-03-01
26	1	2021-11-01
29	1	2099-12-31
30	1	2099-12-31
31	1	2022-05-01

```
[6]: public_firm_match_table = df_position_US[["rcid", "gvkey"]].
      ↪drop_duplicates("rcid")
      public_firm_match_table.rename(columns={'gvkey': 'the_key'}, inplace=True)
```

```
[7]: private_firm_match_table = pd.read_pickle("Processed_Data/
      ↪private_firm_match_table.pkl")
```

```
[13]: firm_match_table = public_firm_match_table.append(private_firm_match_table)
      firm_match_table['rcid'] = firm_match_table['rcid'].str.replace('rcid_', '').
      ↪astype(int)
      firm_match_table.head()
```

```
[13]:      rcid the_key
      25    509865    61676
      46    276621    26144
      55   1037690    63737
      62    786006   183366
      74   20944861   17451
```

```
[14]: firm_match_table.to_csv("Processed_Data/firm_match_table.csv")
```

```
[6]: df_position_US[~df_position_US["gvkey"].notna()].head(100)
```

```
[6]: Empty DataFrame
      Columns: [user_id, position_id, company_cleaned, region, country, state, msa,
      startdate, enddate, jobtitle_raw, salary, seniority, rn, rcid, onet_code, cusip,
      gvkey, len_startdate, d_date, enddate2]
      Index: []
```

```
[5]: df_position_US = df_position_US.sort_values(["user_id", "startdate"])
      df_position_US['gvkey_lag'] = df_position_US.groupby('user_id')['gvkey'].
      ↪shift(1)
      df_position_US['enddate_lag'] = df_position_US.groupby('user_id')['enddate'].
      ↪shift(1)
      df_position_US['d_poaching'] = (
          (df_position_US['gvkey'] != df_position_US['gvkey_lag']) &
          (df_position_US['gvkey_lag'].notna()) &
          (df_position_US['enddate_lag'] == df_position_US['startdate']))
```

```
).astype(int)
```

```
[16]: df_position_US = df_position_US[["user_id", "position_id", "state", "msa",  
↳ "seniority", "startdate", "enddate", "gvkey", "d_poaching"]]
```

```
[17]: df_position_US.head()
```

```
[17]:
```

	user_id	position_id	state	\
29706	user_id_1000000	position_id_-4969643644786663037	NC	
30137	user_id_100000018	position_id_2265739349938949442	CA	
646071	user_id_100000033	position_id_-6229113525456998928	CA	
207850	user_id_100000033	position_id_5628340360971591489	CA	
719712	user_id_100000033	position_id_3182664419104348067	CA	

	msa	seniority	startdate	\
29706	Charlotte-Gastonia-Concord NC-SC MSA	1.0	2012-09-01	
30137	Los Angeles-Long Beach-Santa Ana CA MSA	5	2011-01-01	
646071	Santa Cruz-Watsonville CA MSA	2.0	1997-12-01	
207850	San Jose-Sunnyvale-Santa Clara CA MSA	2.0	1998-06-01	
719712	San Jose-Sunnyvale-Santa Clara CA MSA	3	2000-01-01	

	enddate	gvkey	d_poaching
29706	NaN	2812	0
30137	2013-01-01	2555	0
646071	1998-06-01	126434	0
207850	1999-01-01	6774	1
719712	2001-01-01	115404	0

```
[18]: df_users = pd.read_pickle("Processed_Data/df_users_clean.pkl")  
df_users = df_users[["user_id", "sex_predicted", "ethnicity_predicted"]]  
df_users.head()
```

```
[18]:
```

	user_id	sex_predicted	ethnicity_predicted
1	user_id_100003999	Female	White
12	user_id_100013695	Male	White
15	user_id_100016363	Male	Hispanic
20	user_id_100018128	Male	White
21	user_id_100018206	Female	White

2 Merge everything

```
[19]: df_sum = df_position_US.merge(df_users, on = "user_id", how = "left")
```

```
[21]: df_sum.to_pickle("Processed_Data/df_level_demographics.pkl")
```

3 Create aggregated measure

```
[1]: import pandas as pd
import swifter
pd.set_option('display.max_columns', 500)
pd.set_option('display.max_rows', 500)
import os
os.chdir(r"D:\User.Data\Dan.Li\Revelio")
```

```
[2]: df_sum = pd.read_pickle("Processed_Data/df_level_demographics.pkl")
```

```
[3]: df_sum = df_sum[["seniority", "startdate", "enddate", "gvkey", "sex_predicted", "ethnicity_predicted", "d_poaching"]]
```

```
[4]: df_sum.to_pickle("Processed_Data/df_level_demographics_simple.pkl")
```

```
[6]: df_sum.head()
```

```
[6]:
```

	seniority	startdate	enddate	gvkey	sex_predicted	ethnicity_predicted	\
0	1.0	2012-09-01	NaN	2812	Female	White	
1	5	2011-01-01	2013-01-01	2555	Female	White	
2	2.0	1997-12-01	1998-06-01	126434	Male	White	
3	2.0	1998-06-01	1999-01-01	6774	Male	White	
4	3	2000-01-01	2001-01-01	115404	Male	White	

	d_poaching
0	0
1	0
2	0
3	1
4	0

```
[7]: df_sum['enddate'] = df_sum['enddate'].fillna(pd.to_datetime('2099-12-31'))
df_sum['startdate'] = pd.to_datetime(df_sum['startdate']).dt.date
df_sum['enddate'] = pd.to_datetime(df_sum['enddate']).dt.date
df_sum = df_sum[df_sum["startdate"].notna()]
```

4 Create panel

```
[11]: id_list = df_sum.drop_duplicates("gvkey")["gvkey"].to_list()
```

```
[12]: import pandas as pd
from itertools import product

# Generating year-month combinations from 1980-01 to 2022-12
date_range = pd.date_range(start='2018-01-01', end='2022-12-01', freq='MS')
```

```

year_month_combinations = [(date.year, date.month) for date in date_range]

# Creating a list of dictionaries containing all combinations of
# factset_entity_id, year, month, and date
panel_data = []
for entity_id in id_list:
    for year, month in year_month_combinations:
        date = pd.Timestamp(year=year, month=month, day=1)
        panel_data.append({
            'gvkey': entity_id,
            'year': year,
            'month': month,
            'date': date
        })

# Creating a DataFrame from the list of dictionaries
panel_df = pd.DataFrame(panel_data)

# Displaying the first few rows of the panel DataFrame
print(panel_df.head())

```

	gvkey	year	month	date
0	2812	2018	1	2018-01-01
1	2812	2018	2	2018-02-01
2	2812	2018	3	2018-03-01
3	2812	2018	4	2018-04-01
4	2812	2018	5	2018-05-01

```

[13]: import pandas as pd
import numpy as np
from tqdm import tqdm

# Split panel_df into 100 chunks based on index
num_chunks = 100
panel_indices = np.array_split(panel_df.index, num_chunks)

result_batches = []

# Process each chunk using tqdm for progress visualization
for indices in tqdm(panel_indices, desc="Processing chunks", unit=" chunk"):
    batch = panel_df.loc[indices]
    panel_df2 = pd.merge(batch, df_sum, on="gvkey")
    panel_df2 = panel_df2[
        (panel_df2["date"] >= panel_df2["startdate"]) &
        (panel_df2["date"] <= panel_df2["enddate"])
    ]
    result_batches.append(panel_df2)

```

```
# Concatenate the results
final_result = pd.concat(result_batches)
```

```
Processing chunks: 100%|
100/100 [1:30:01<00:00, 54.01s/ chunk]
```

```
[14]: final_result.head()
```

```
[14]:
```

	gvkey	year	month	date	seniority	startdate	enddate	\
0	2812	2018	1	2018-01-01	1.0	2012-09-01	2099-12-31	
2	2812	2018	1	2018-01-01	3.0	2003-06-01	2019-02-01	
4	2812	2018	1	2018-01-01	4	2015-04-01	2021-02-01	
10	2812	2018	1	2018-01-01	3.0	2017-08-01	2022-04-01	
16	2812	2018	1	2018-01-01	1	2013-07-01	2018-12-01	

	sex_predicted	ethnicity_predicted	d_poaching
0	Female	White	0
2	Male	White	0
4	Female	White	0
10	Male	White	0
16	empty	API	0

```
[16]: final_result.to_pickle("Processed_Data/df_level_demographics_simple_panel18_22.
    ↪pk1")
```

5 Process Panel

```
[1]: import pandas as pd
import swifter
pd.set_option('display.max_columns', 500)
pd.set_option('display.max_rows', 500)
import os
os.chdir(r"D:\User.Data\Dan.Li\Revelio")
```

```
[2]: panel_df2 = pd.read_pickle("Processed_Data/
    ↪df_level_demographics_simple_panel18_22.pkl")
```

```
[7]: panel_df2 = panel_df2[panel_df2["year"]>=2018]
```

```
[8]: panel_df2 = panel_df2.reset_index()
```

```
[10]: panel_df2["d_male"] = panel_df2["sex_predicted"].swifter.
    ↪allow_dask_on_strings(enable=True).apply(lambda x:1 if x == "Male" else 0)
```

```
Dask Apply: 0%|
| 0/80 [00:00<?, ?it/s]
```

```
[11]: panel_df2["d_female"] = panel_df2["sex_predicted"].swifter.
      ↪allow_dask_on_strings(enable=True).apply(lambda x:1 if x == "Female" else 0)
panel_df2["d_api"] = panel_df2["ethnicity_predicted"].swifter.
      ↪allow_dask_on_strings(enable=True).apply(lambda x:1 if x == "API" else 0)
panel_df2["d_white"] = panel_df2["ethnicity_predicted"].swifter.
      ↪allow_dask_on_strings(enable=True).apply(lambda x:1 if x == "White" else 0)
panel_df2["d_hispanic"] = panel_df2["ethnicity_predicted"].swifter.
      ↪allow_dask_on_strings(enable=True).apply(lambda x:1 if x == "Hispanic" else 0)
      ↪0)
panel_df2["d_black"] = panel_df2["ethnicity_predicted"].swifter.
      ↪allow_dask_on_strings(enable=True).apply(lambda x:1 if x == "Black" else 0)
```

Dask Apply: 0%| | 0/80 [00:00<?, ?it/s]

Dask Apply: 0%| | 0/80 [00:00<?, ?it/s]

Dask Apply: 0%| | 0/80 [00:00<?, ?it/s]

Dask Apply: 0%| | 0/80 [00:00<?, ?it/s]

Dask Apply: 0%| | 0/80 [00:00<?, ?it/s]

```
[12]: panel_df2.head()
```

```
[12]:   index  gvkey  year  month      date seniority  startdate  enddate  \
0      0   2812  2018      1 2018-01-01        1.0  2012-09-01  2099-12-31
1      2   2812  2018      1 2018-01-01        3.0  2003-06-01  2019-02-01
2      4   2812  2018      1 2018-01-01         4  2015-04-01  2021-02-01
3     10   2812  2018      1 2018-01-01        3.0  2017-08-01  2022-04-01
4     16   2812  2018      1 2018-01-01         1  2013-07-01  2018-12-01
```

```
      sex_predicted ethnicity_predicted  d_poaching  d_male  d_female  d_api  \
0          Female             White              0      0          1      0
1           Male             White              0      1          0      0
2          Female             White              0      0          1      0
3           Male             White              0      1          0      0
4         empty              API              0      0          0      1
```

```
      d_white  d_hispanic  d_black
0          1           0         0
1          1           0         0
2          1           0         0
3          1           0         0
4          0           0         0
```

```
[13]: import swifter
      def convert_to_int(x):
          try:
              x = int(x)
```

```

        if x>=4:
            return 1
        else:
            return 0

    except (ValueError, TypeError):
        return np.nan

```

```
[15]: import numpy as np
```

```
[16]: panel_df2["d_senior"] = panel_df2["seniority"].swifter.progress_bar(True).
      ↪allow_dask_on_strings(True).apply(convert_to_int)
```

Dask Apply: 0%| | 0/80 [00:00<?, ?it/s]

```
[17]: # Columns for interaction (replace with specific column names)
      columns_for_interaction = ['d_male', 'd_female', 'd_api', 'd_white',
      ↪ 'd_hispanic', 'd_black']

      for col in columns_for_interaction:
          panel_df2[col+'_senior'] = panel_df2['d_senior'] * panel_df2[col]

```

```
[21]: df_aggregated = panel_df2.groupby(["gvkey", "date", "year", "month"]).sum().
      ↪reset_index()
```

```
[22]: # Calculate gender_measurable and race_measurable
      df_aggregated['gender_measurable'] = df_aggregated['d_male'] +
      ↪df_aggregated['d_female']
      df_aggregated['race_measurable'] = df_aggregated['d_api'] +
      ↪df_aggregated['d_white'] + df_aggregated['d_hispanic'] +
      ↪df_aggregated['d_black']

      # Calculate ratios
      df_aggregated['male_ratio'] = df_aggregated['d_male'] /
      ↪df_aggregated['gender_measurable']
      df_aggregated['female_ratio'] = df_aggregated['d_female'] /
      ↪df_aggregated['gender_measurable']

      df_aggregated['api_ratio'] = df_aggregated['d_api'] /
      ↪df_aggregated['race_measurable']
      df_aggregated['white_ratio'] = df_aggregated['d_white'] /
      ↪df_aggregated['race_measurable']
      df_aggregated['hispanic_ratio'] = df_aggregated['d_hispanic'] /
      ↪df_aggregated['race_measurable']
      df_aggregated['black_ratio'] = df_aggregated['d_black'] /
      ↪df_aggregated['race_measurable']

```

```

# Calculate gender_measurable_senior and race_measurable_senior
df_aggregated['gender_measurable_senior'] = df_aggregated['d_male_senior'] +
↳ df_aggregated['d_female_senior']
df_aggregated['race_measurable_senior'] = df_aggregated['d_api_senior'] +
↳ df_aggregated['d_white_senior'] + df_aggregated['d_hispanic_senior'] +
↳ df_aggregated['d_black_senior']

# Calculate ratios for senior demographic variables
df_aggregated['male_senior_ratio'] = df_aggregated['d_male_senior'] /
↳ df_aggregated['gender_measurable_senior']
df_aggregated['female_senior_ratio'] = df_aggregated['d_female_senior'] /
↳ df_aggregated['gender_measurable_senior']

df_aggregated['api_senior_ratio'] = df_aggregated['d_api_senior'] /
↳ df_aggregated['race_measurable_senior']
df_aggregated['white_senior_ratio'] = df_aggregated['d_white_senior'] /
↳ df_aggregated['race_measurable_senior']
df_aggregated['hispanic_senior_ratio'] = df_aggregated['d_hispanic_senior'] /
↳ df_aggregated['race_measurable_senior']
df_aggregated['black_senior_ratio'] = df_aggregated['d_black_senior'] /
↳ df_aggregated['race_measurable_senior']

```

```

[25]: columns_to_prefix = [
    'd_male', 'd_female', 'd_api', 'd_white', 'd_hispanic', 'd_black',
    'gender_measurable', 'race_measurable', 'male_ratio', 'female_ratio',
    'api_ratio', 'white_ratio', 'hispanic_ratio', 'black_ratio',
    'd_male_senior', 'd_female_senior', 'd_api_senior', 'd_white_senior',
↳ 'd_hispanic_senior', 'd_black_senior',
    'gender_measurable_senior', 'race_measurable_senior', 'male_senior_ratio',
    'female_senior_ratio', 'api_senior_ratio', 'white_senior_ratio',
↳ 'hispanic_senior_ratio', 'black_senior_ratio'
]

for column in columns_to_prefix:
    df_aggregated.rename(columns={column: 'l_' + column}, inplace=True)

```

```

[27]: df_aggregated.to_csv("Processed_Data/20240824df_level_aggre_bymonth.csv")

```

```
[ ]:
```

```
[ ]:
```