OBJECTIVE OF ANALYSIS

The major objective of the analysis is to visualize salaries based on different metrics like gender, experience, and education for the IT industry in United States of America from the Survey dataset.

RESEARCH QUESTION

Are male software engineers paid higher salaries than their female and non-binary counterparts ?

MOTIVATION FOR CHOOSING THIS DATASET

This dataset consists of various types of inputs and inconsistent inputs that makes it ideal for data cleaning practice.

IMPORTING NECESSARY LIBRARIES

In [ ]:

```python
import pandas as pd
import numpy as np
import difflib
import fuzzywuzzy
from fuzzywuzzy import process
import seaborn as sns
import matplotlib.pyplot as plt
import chardet
import openpyxl
```

```
/Users/anshumangupta/Desktop/portfolio/Data Cleaning/Data Cleaning/salary/lib/
python3.10/site-packages/fuzzywuzzy/fuzz.py:11: UserWarning: Using slow pure-p
ython SequenceMatcher. Install python-Levenshtein to remove this warning
  warnings.warn('Using slow pure-python SequenceMatcher. Install python-Levens
htein to remove this warning')
```

IMPORTING DATA (.csv)

In [ ]:

```python
# Loading raw data into pandas dataframe
raw_data = pd.DataFrame(pd.read_csv('data/survey.csv'))

# Viewing first 5 rows
raw_data.head(5)
```

Out[ ]:

	Timestamp	How old are you?	What industry do you work in?	Job title	If your job title needs additional context, please clarify here:	What is your annual salary? (You'll indicate the currency in a later question. If you are part-time or hourly, please enter an annualized equivalent -- what you would earn if you worked the job 40 hours a week, 52 weeks a year.)	How much additional monetary compensation do you get, if any (for example, bonuses or overtime in an average year)? Please only include monetary compensation here, not the value of benefits.	Plea indic curre
0	4/27/2021 11:02:10	25-34	Education (Higher Education)	Research and Instruction Librarian	NaN	55,000	0.0	U
1	4/27/2021 11:02:22	25-34	Computing or Tech	Change & Internal Communications Manager	NaN	54,600	4000.0	G
2	4/27/2021 11:02:38	25-34	Accounting, Banking & Finance	Marketing Specialist	NaN	34,000	NaN	U
3	4/27/2021 11:02:41	25-34	Nonprofits	Program Manager	NaN	62,000	3000.0	U
4	4/27/2021 11:02:42	25-34	Accounting, Banking & Finance	Accounting Manager	NaN	60,000	7000.0	U

CHECKING NUMBER OF ROWS AND COLUMNS IN THE DATASET

In [ ]:

```
# Checking number of rows and columns
raw_data.shape
```

Out[ ]:

```
(27893, 18)
```

CHANGING COLUMN HEADERS FOR SIMPLICITY

In [ ]:

```
# Setting new column headers
new_column_headers = ['timestamp','age_group','curr_industry','job_title','jo
init_data = raw_data.set_axis(new_column_headers,axis=1,inplace=False)
init_data.head(5)
```

```
/var/folders/h2/3d9l8ncd5xdgj1h3ffmkqykw0000gn/T/ipykernel_39323/4246118180.p
y:3: FutureWarning: DataFrame.set_axis 'inplace' keyword is deprecated and wil
l be removed in a future version. Use `obj = obj.set_axis(..., copy=False)` in
stead
  init_data = raw_data.set_axis(new_column_headers,axis=1,inplace=False)
```

Out[ ]:

	timestamp	age_group	curr_industry	job_title	job_context	curr_salary	compensati
0	4/27/2021 11:02:10	25-34	Education (Higher Education)	Research and Instruction Librarian	NaN	55,000	
1	4/27/2021 11:02:22	25-34	Computing or Tech	Change & Internal Communications Manager	NaN	54,600	400
2	4/27/2021 11:02:38	25-34	Accounting, Banking & Finance	Marketing Specialist	NaN	34,000	N
3	4/27/2021 11:02:41	25-34	Nonprofits	Program Manager	NaN	62,000	300
4	4/27/2021 11:02:42	25-34	Accounting, Banking & Finance	Accounting Manager	NaN	60,000	700

REMOVING UNNECCESARY DATA

> > The research questions are not influenced by attributes like Race, Timestamp, Additional Currencies, Cities. Therefore, they were removed from the final dataset.

In [ ]:

```
# Dropping unnecessary columns
init_data.drop(columns=["job_context","compensation","income_context","race",
```

In [ ]:

```
# Converting curr_salary column to numeric data type
init_data['curr_salary'] = init_data['curr_salary'].str.replace(",","",regex=
init_data['curr_salary'] = pd.to_numeric(init_data['curr_salary'])

init_data
```

Out[ ]:

	age_group	curr_industry	job_title	curr_salary	currency	country	state_U
0	25-34	Education (Higher Education)	Research and Instruction Librarian	55000	USD	United States	Massachuse
1	25-34	Computing or Tech	Change & Internal Communications Manager	54600	GBP	United Kingdom	N

	age_group	curr_industry	job_title	curr_salary	currency	country	state_U
2	25-34	Accounting, Banking & Finance	Marketing Specialist	34000	USD	US	Tenness
3	25-34	Nonprofits	Program Manager	62000	USD	USA	Wiscon
4	25-34	Accounting, Banking & Finance	Accounting Manager	60000	USD	US	South Caroli
...	...	...	...	...	...	...	
27888	25-34	Computing or Tech	Systems Engineer	114000	USD	US	Wiscon
27889	18-24	Business or Consulting	Data Management Consultant	60000	GBP	United Kingdom	N
27890	35-44	Engineering or Manufacturing	Project Engineer	75000	USD	United States	Misso
27891	18-24	Computing or Tech	Technology Associate	8600	EUR	Romania	N
27892	35-44	Computing or Tech	Program Manager	138000	USD	USA	Tex

27893 rows × 12 columns

CONSOLIDATING ALL COUNTRY COLUMN VALUES

In [ ]:

```
# Count of unique values in country column
print(len(init_data.country.unique()))
```

```
368
```

In [ ]:

```
# The script below removes whitespaces from the values in the country columns

countries_list = ['us', 'statesofamerica', 'america', 'usa', 'unitedstates']


init_data['country'] = init_data['country'].str.lower()
init_data['country'] = init_data['country'].str.replace('\W', '')
init_data['state_USA'] = init_data['state_USA'].replace(np.nan,'N/A')

# checking values for every column to get top match with 70% match rate
for i in init_data.index:
    country = init_data.at[i,"country"]
    matched_country = difflib.get_close_matches(country,countries_list,n=1,cu

    if matched_country:
        init_data.at[i,'country'] = matched_country[0]

# converting values of matched countries to 'us'
for i in init_data.index:
    if init_data.at[i,"country"] in countries_list:
        init_data.at[i,"country"] = 'us'
```

```
/var/folders/h2/3d9l8ncd5xdgj1h3ffmkqykw0000gn/T/ipykernel_39323/1979729213.p
y:7: FutureWarning: The default value of regex will change from True to False
in a future version.
  init_data['country'] = init_data['country'].str.replace('\W', '')
```

In [ ]:

```
# Checking the unique values in the country column to verify
print(len(init_data.country.unique()))
```

```
203
```

CONSOLIDATING GENDER COLUMN

In [ ]:

```
# Checking NaNs in the column
print(init_data["gender"].isna().sum())

# Checking unique values for gender column
print(init_data["gender"].unique())
```

```
165
['Woman' 'Non-binary' 'Man' nan 'Other or prefer not to answer'
 'Prefer not to answer']
```

In [ ]:

```
# Setting Man, Woman, and Non-Binary to M,F,NB respectively
init_data.loc[(init_data['gender']=='Man'), 'gender'] = 'M'
init_data.loc[(init_data['gender']=='Woman'), 'gender'] = 'F'
init_data.loc[(init_data['gender']=='Non-binary'), 'gender'] = 'NB'

# Setting "Other or prefer not to answer" to Other as the value "Prefer not t
# Survey participants can identify as other genders that were not part of the
init_data.loc[(init_data['gender']=='Other or prefer not to answer'), 'gender

# Setting NaN to No Answer
init_data["gender"].fillna("No Answer", inplace=True)

# Checking NaNs in the column
print(init_data["gender"].isna().sum())
```

```
0
```

CONSOLIDATING VALUES FOR CURRENT INDUSTRY

In [ ]:

```
print(len(init_data["curr_industry"].unique()))
print(init_data["curr_industry"].isna().sum())
# Dropping NaN values
init_data.dropna(inplace=True)
print(init_data["curr_industry"].isna().sum())
```

```
1210
71
0
```

APPLYING FUZZY MATCHING TO BRING VALUES LIKE ["Software" , "Programming", "Computing"] into IT.

In [ ]:

```
# removing trailing whitespaces
init_data["curr_industry"] = init_data["curr_industry"].str.strip()

# chanign to lowercase
init_data["curr_industry"] = init_data["curr_industry"].str.lower()


# Making function for fuzzy matching
def fuzzy_match(df, col, text_to_match, text_to_replace, ratio, lim=10):

    # Getting unique values from input column
    unique_ind = df[col].unique()

    # Running fuzzy matching algorithm from fuzzywuzzy library
    matched_ind = fuzzywuzzy.process.extract(text_to_match,unique_ind, limit=

    # If the similarity ratio is less than the given ratio
    if matched_ind[0][1] < ratio:
        print(f"Ratio given is too high. Try less than or equal to {matched_i
    else:
        # Create an array for best matches
        best_match = [match[0] for match in matched_ind if match[1]>=ratio]
        i = 0
        print("Matches returned :")
        # Print matches to see if the output is correct
        while best_match!=[] and i<len(best_match):
            print(best_match[i])
            i+=1
        # Giving user an option to proceed or abort the matching
        ch = input("Check matches and press 'Y' to change (press and other ke

        # Proceeding with replacement if selection is "y"
        if ch.lower()=="y":

            # creating another column that returns boolean values for matches
            to_replace = df[col].isin(best_match)

            # replacing where boolean values are True
            df.loc[(to_replace,col)] = text_to_replace
            print(f"Replaced {text_to_match} instances in {col} column with {
    return




fuzzy_match(init_data, "curr_industry" ,"computing software", "IT", 59)
```

```
Matches returned :
computing or tech
software/programming
saas company/software
strategy consulting
software development / it
software development
software products
software
payroll software
biotech/software
Replaced computing software instances in curr_industry column with IT | unique
values are : 983
```

Validating results

In [ ]:

```
init_data.head(10)
```

Out[ ]:

	age_group	curr_industry	job_title	curr_salary	currency	country	sta
0	25-34	education (higher education)	Research and Instruction Librarian	55000	USD	us	Massa
1	25-34	IT	Change & Internal Communications Manager	54600	GBP	unitedkingdom	
2	25-34	accounting, banking & finance	Marketing Specialist	34000	USD	us	Te
3	25-34	nonprofits	Program Manager	62000	USD	us	W
4	25-34	accounting, banking & finance	Accounting Manager	60000	USD	us	South
5	25-34	education (higher education)	Scholarly Publishing Librarian	62000	USD	us	Ha
6	25-34	publishing	Publishing Assistant	33000	USD	us	South
7	25-34	education (primary/secondary)	Librarian	50000	USD	us	
8	45-54	IT	Systems Analyst	112000	USD	us	
9	35-44	accounting, banking & finance	Senior Accountant	45000	USD	us	

Modifying dataset so that all values are for United States for the IT industry

In [ ]:

```
us_dataset = init_data.loc[(init_data['country']== 'us') & (init_data['curr_i

# Dropping cities as it is not relevant to research questions
us_dataset = us_dataset.drop('city',axis=1)
```

In [ ]:

```
# Resetting index
us_dataset = us_dataset.reset_index(drop=True)
```

In [ ]:

```
# Sorting values
us_dataset.sort_values(by=['curr_salary'], inplace=True)
us_dataset.head(10)
```

Out[ ]:

	age_group	curr_industry	job_title	curr_salary	currency	country	state_USA	ove
1628	18-24	IT	Product Marketer	0	USD	us	California	
2595	45-54	IT	Founder	0	USD	us	California	

	age_group	curr_industry	job_title	curr_salary	currency	country	state_USA	ove
852	35-44	IT	Software Development Lead	1	USD	us	Wisconsin	8 -
1201	45-54	IT	Account Manager	55	USD	us	New Hampshire	
2180	25-34	IT	Technical Writer	72	USD	us	Washington	8 -
2264	45-54	IT	Coach	130	USD	us	N/A	
1730	25-34	IT	Chief Data Scientist	240	USD	us	California	5
1038	25-34	IT	Sr Consultant	10000	USD	us	District of Columbia	8 -
3546	35-44	IT	Software Engineer Technical Support	10700	USD	us	Texas	8 -
819	35-44	IT	Manager of Customer Support	13000	USD	us	California	

In [ ]:

```
# Converting Job titles to lower case
us_dataset['job_title'] = us_dataset['job_title'].str.lower()
```

In [ ]:

```
#software engineer dataset

# The line of code below extracts all job title swith 'software engineer'
# in the string and leaves out all job titles with senior, lead, principle, o

se_ds = us_dataset.loc[(us_dataset['job_title'].str.contains('software engine
```

In [ ]:

```
se_ds
```

Out[ ]:

	age_group	curr_industry	job_title	curr_salary	currency	country	state
1990	25-34	IT	associate software engineer	57000	USD	us	I
167	25-34	IT	software engineer	60000	USD	us	Cal
649	18-24	IT	software engineer in test	60000	USD	us	Ne
2731	25-34	IT	software engineer	61000	USD	us	Cal
1690	35-44	IT	developer (software engineer/programmer)	63500	USD	us	Mi
...	...	...	...	...	...	...	
3561	25-34	IT	software engineer	340000	USD	us	Mic

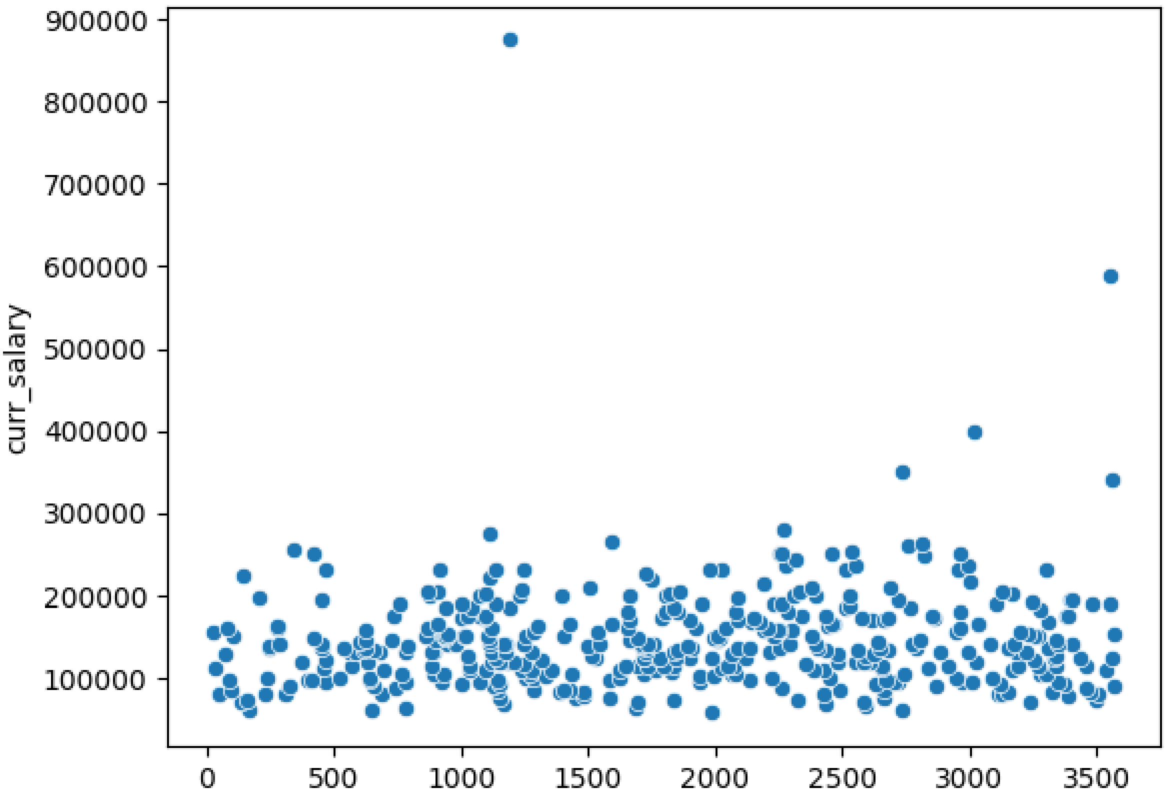
	age_group	curr_industry	job_title	curr_salary	currency	country	state
							Wash
2738	25-34	IT	software engineer	350000	USD	us	Cal
3014	25-34	IT	software engineer	400000	USD	us	Massach
3551	25-34	IT	software engineer	590000	USD	us	Ne
1191	25-34	IT	software engineer	875000	USD	us	Cal

406 rows × 11 columns

Searching for outliers

In [ ]:

```
# Creating scatter plot for outliers
sns.scatterplot(y=se_ds['curr_salary'],x=se_ds.index)
plt.show()
```



Most of the vlaues are concentrated around 100,000 and around 250,000

In [ ]:

```
# Removing outliers from the dataset using Inter-Quartile Range (IQR)
# IQR depicts the spread of values in the current salary column

qt_1, qt_3 = np.percentile(se_ds['curr_salary'],[25,75])
iqr = qt_3-qt_1
lower_bound = qt_1 - (1.5 * iqr)
upper_bound = qt_3 + (1.5 * iqr)
```

```
print(lower_bound, upper_bound)

se_ds = se_ds.loc[(se_ds['curr_salary'] > lower_bound) & (se_ds['curr_salary'
```

```
16250.0 266250.0
```

In [ ]:

```
# resetting index
se_ds = se_ds.reset_index(drop=True)
```

In [ ]:

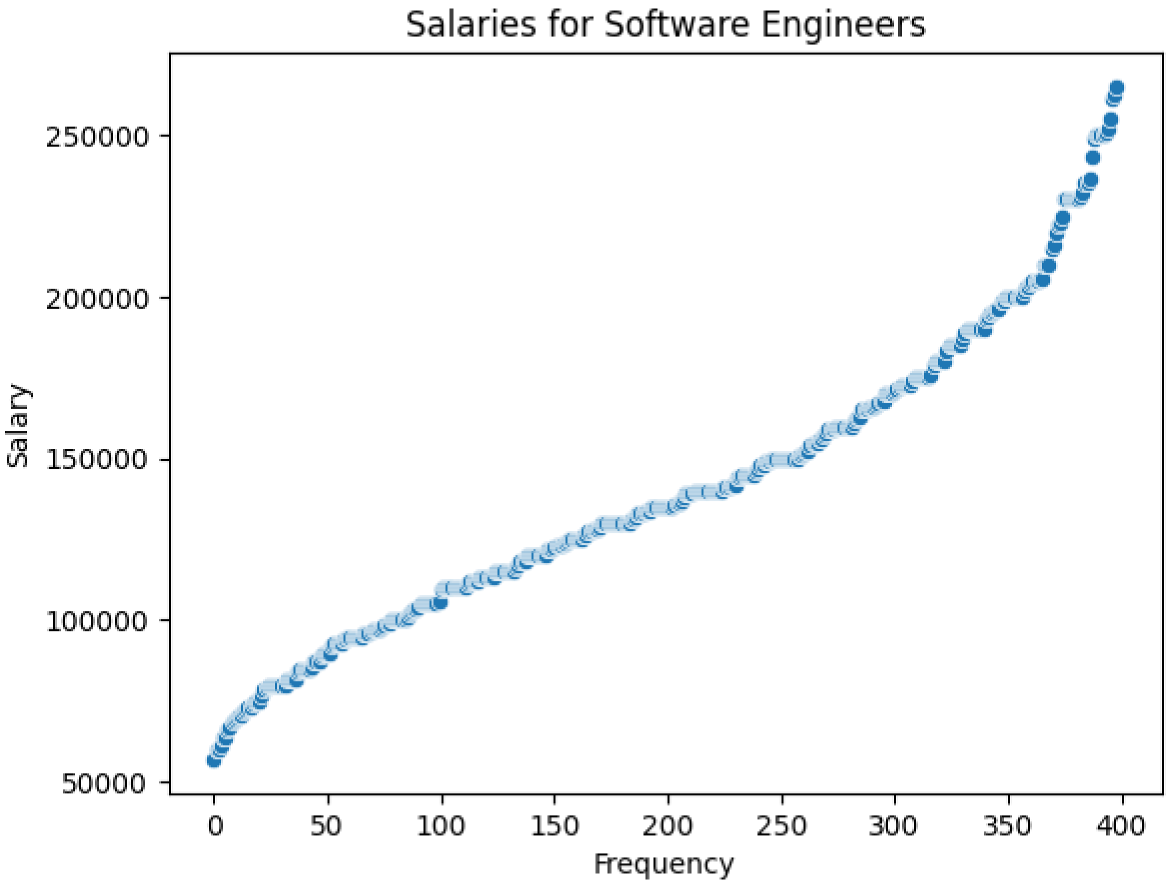
```
# Describing dataset
se_ds.describe()
```

Out[ ]:

	curr_salary
count	399.000000
mean	140925.408521
std	45374.261953
min	57000.000000
25%	107725.000000
50%	135000.000000
75%	170200.000000
max	265000.000000

In [ ]:

```
sns.scatterplot(y=se_ds['curr_salary'],x=se_ds.index)
plt.title("Salaries for Software Engineers")
plt.xlabel('Frequency')
plt.ylabel('Salary')
plt.show()
```

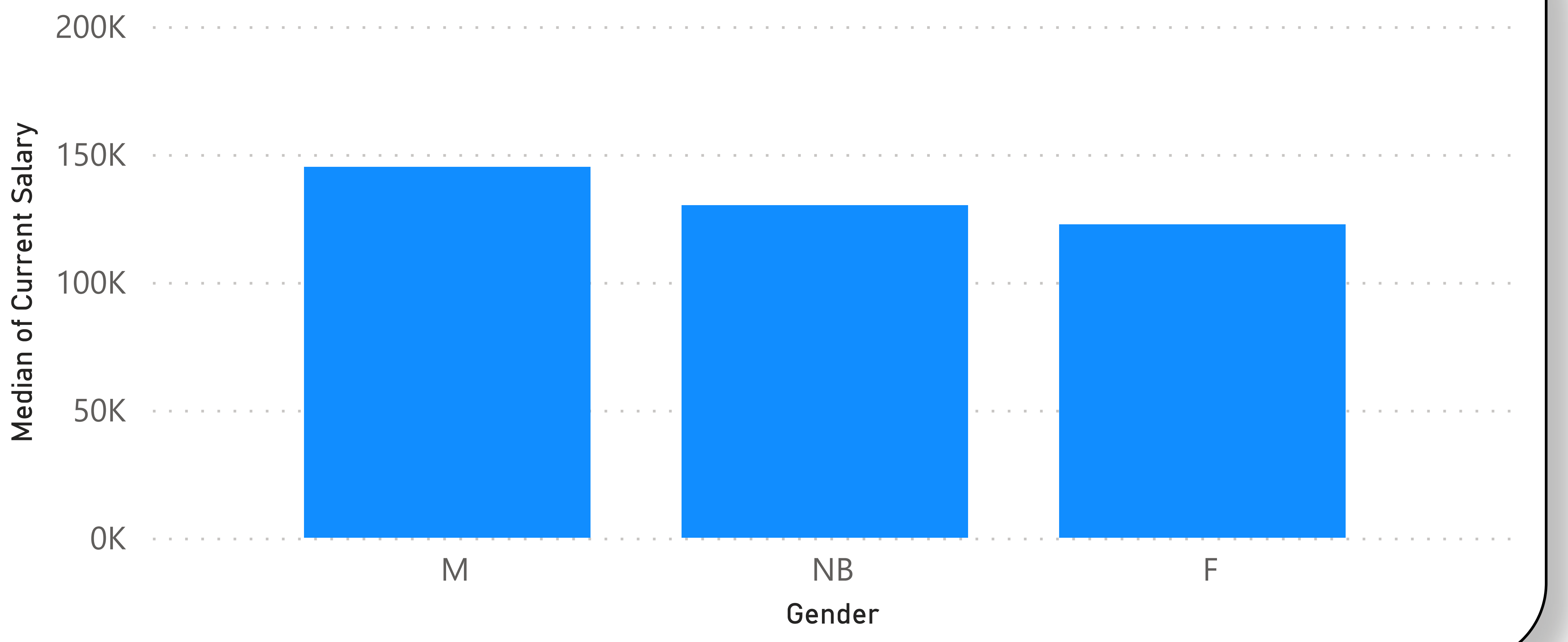


In [ ]:

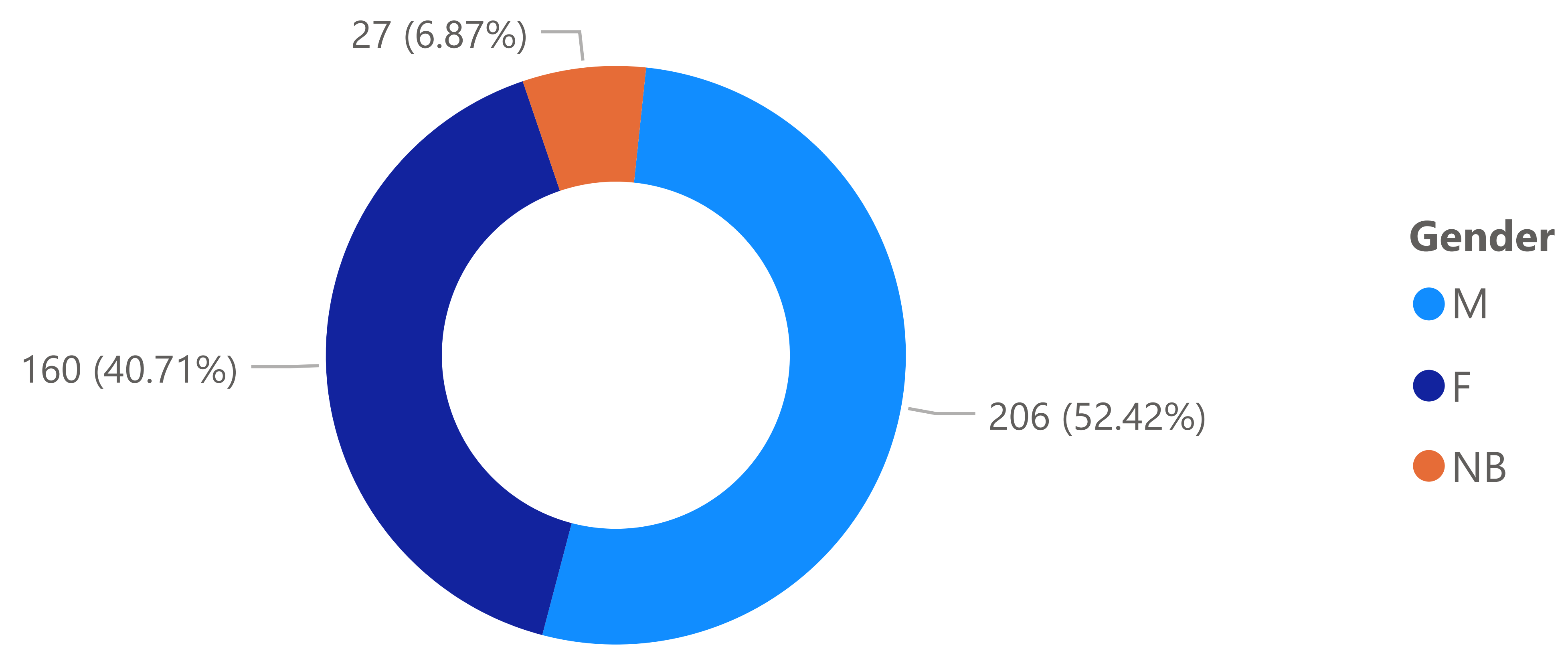
```
# Exporting to Excel for Visualization in Power BI

filename = 'se_dataset.xlsx'
se_ds.to_excel(filename)
```

In [ ]:

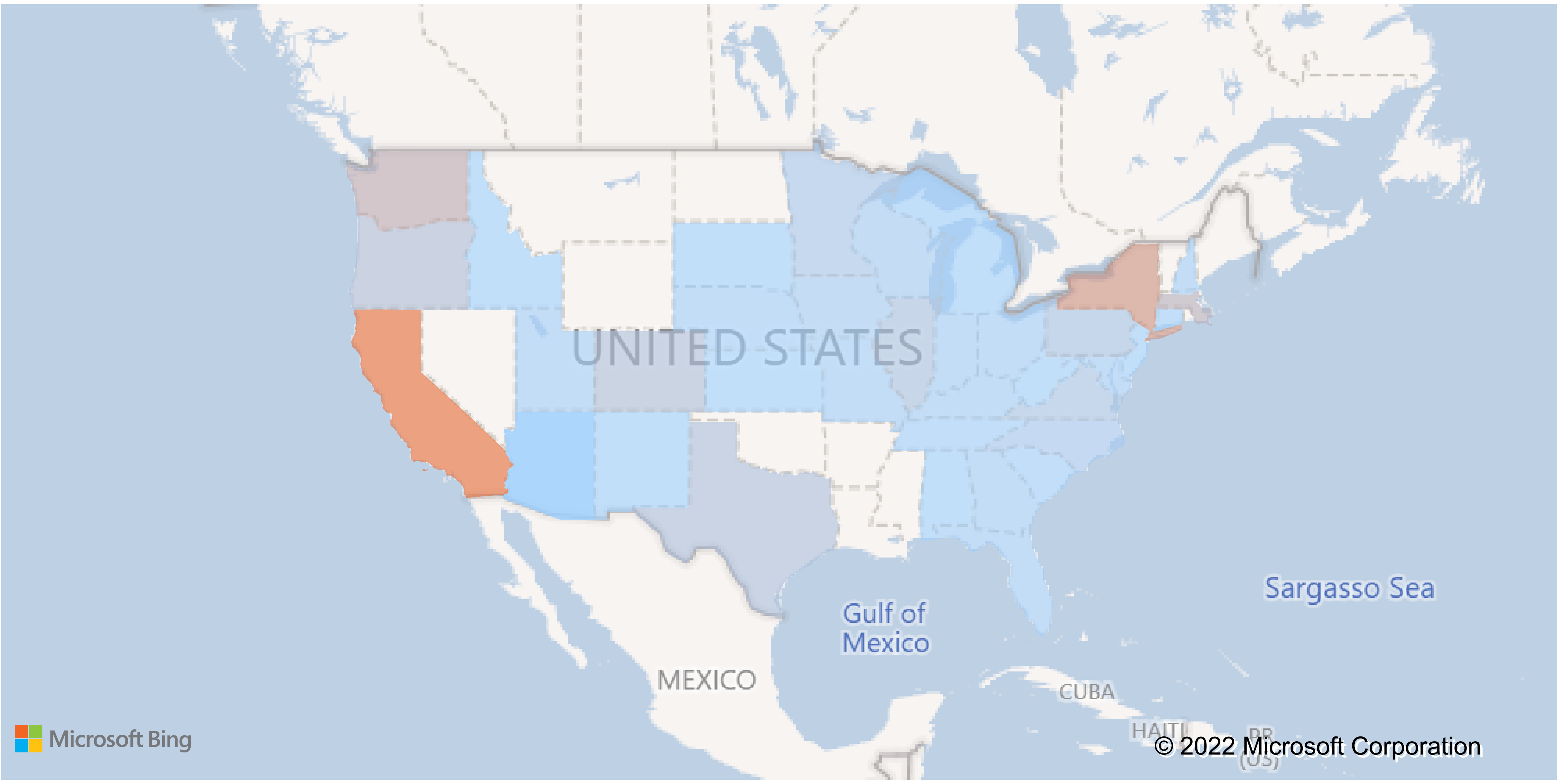
Median Salary by Gender



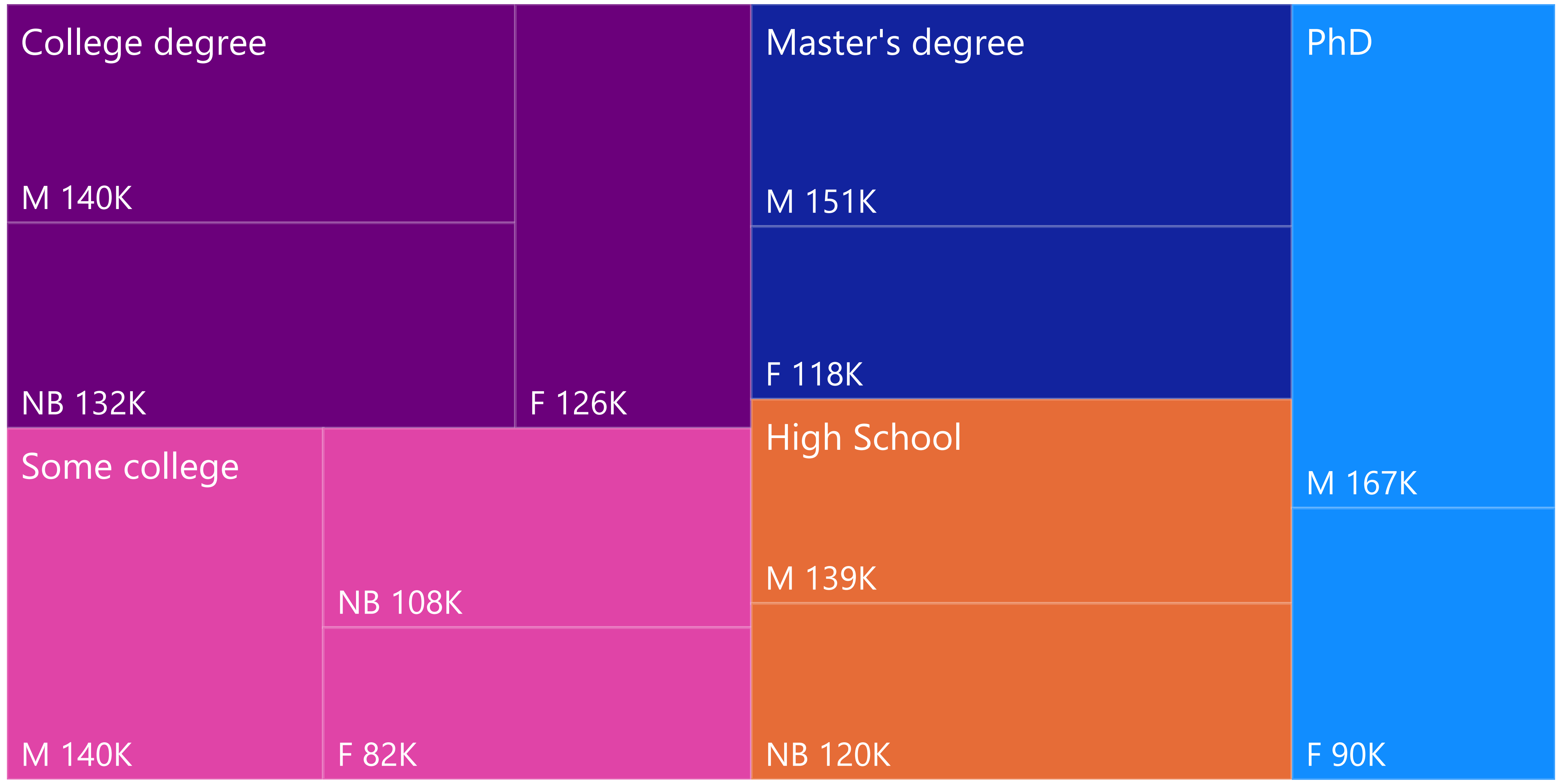
Gender Composition



State (in US)



Median of Current Salary by Education and Gender


College degree
M 140K
NB 132K
F 126K
Master's degree
M 151K
F 118K
PhD
M 167K
F 90K
Some college
M 140K
NB 108K
F 82K
High School
M 139K
NB 120K