Salary_cleaning

February 21, 2023

[211]:

```python
# import necessary libraries - (CELL 1)
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from fuzzywuzzy import fuzz
```

[212]:

```python
# read in the dataset - (CELL 2)
data = pd.read_csv("Surveys.csv", sep=',').replace('"', '', regex=True)
```

[213]:

```python
# display data to get a grasp on what the whole dataset looks like - (CELL 3)
display(data.head())
```

```
            Timestamp How old are you?  What industry do you work in?  \
0  4/27/2021 11:02:10            25-34   Education (Higher Education)
1  4/27/2021 11:02:22            25-34              Computing or Tech
2  4/27/2021 11:02:38            25-34  Accounting, Banking & Finance
3  4/27/2021 11:02:41            25-34                     Nonprofits
4  4/27/2021 11:02:42            25-34  Accounting, Banking & Finance

                                  Job title  \
0        Research and Instruction Librarian
1  Change & Internal Communications Manager
2                      Marketing Specialist
3                           Program Manager
4                        Accounting Manager

  If your job title needs additional context, please clarify here:  \
0                                              NaN
1                                              NaN
2                                              NaN
3                                              NaN
4                                              NaN

  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.)  \
0                                             55,000
1                                             54,600
```

```
2                                             34,000
3                                             62,000
4                                             60,000

   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.  \
0                                                0.0
1                                             4000.0
2                                                NaN
3                                             3000.0
4                                             7000.0

  Please indicate the currency  \
0                          USD
1                          GBP
2                          USD
3                          USD
4                          USD

  If "Other," please indicate the currency here:   \
0                                             NaN
1                                             NaN
2                                             NaN
3                                             NaN
4                                             NaN

  If your income needs additional context, please provide it here:  \
0                                                NaN
1                                                NaN
2                                                NaN
3                                                NaN
4                                                NaN

  What country do you work in?  \
0                United States
1               United Kingdom
2                           US
3                          USA
4                           US

  If you're in the U.S., what state do you work in? What city do you work in?  \
0                                     Massachusetts                    Boston
1                                               NaN                 Cambridge
2                                         Tennessee               Chattanooga
3                                         Wisconsin                 Milwaukee
4                                    South Carolina                Greenville
```

```
  How many years of professional work experience do you have overall?  \
0                                          5-7 years
1                                       8 - 10 years
2                                        2 - 4 years
3                                       8 - 10 years
4                                       8 - 10 years

  How many years of professional work experience do you have in your field?  \
0                                          5-7 years
1                                          5-7 years
2                                        2 - 4 years
3                                          5-7 years
4                                          5-7 years

  What is your highest level of education completed? What is your gender?  \
0                                    Master's degree               Woman
1                                     College degree          Non-binary
2                                     College degree               Woman
3                                     College degree               Woman
4                                     College degree               Woman

  What is your race? (Choose all that apply.)
0                                       White
1                                       White
2                                       White
3                                       White
4                                       White
```

[ ]:
```
# (CELL 4)
```

1 Data explanation

The dataset consists of survey data from 2021 and 2022 regarding people's salary. It containts 27922 entries and is made out of 18 variables such as salary, job title, industry, etc. The column names are the questions given to the respondent. The survey data can retrieved from https://oscarbaruffa.com/messy/. The survey form can be retrieved from https://www.askamanager.org/2021/04/how-much-money-do-you-make-4.html.

The following variables are present in the dataset.

1.1 Timestamp

- The datetime at which the respondent submitted their entry.
- This variable was generated by the software handling the survey data
- This variable is never emtpy.

1.2 Age band

- The age band in which the resondent belongs to.
- The respondent chose from a premade list of answers.
- This variable is mandatory.
- Only 1 answer can be chosen.

1.3 Industry

- The industry in which the respondent works.
- The respondent either chose from a premade list of answers or gave their own answer.
- This variable is not mandatory.
- Only 1 answer could be chosen or given.

1.4 Job title

- The job title of the respondent.
- The respondent gave their own answer.
- This variable is mandatory.
- Only 1 answer can given.

1.5 Job context

- Additional context regarding the respondent's job title.
- The respondent gave their own answer.
- This variable is not mandatory.
- Only 1 answer could be given.

1.6 Salary

- The respondent's annual salary based on 40 hours a week, 52 weeks a year.
- The respondent gave their own answer.
- This variable is mandatory.
- Only 1 answer can be given.

1.7 Compensation

- Additional monetary income if the respondent has any.
- The respondent gave their own answer.
- This variable is not mandatory.
- Only 1 answer could be given.

1.8 Currency

- The currency in which the respondent receives their salary abd compensation.
- The respondent chose from a premade list of answers.
- This variable is mandatory ('Other' is an answer in the list).
- Only 1 answer can be chosen.

1.9 Other currency

- The respondent's currency in case it wasn't an option in the premade list.
- The respondent gave their own answer.
- This variable is not mandatory.
- Only 1 answer could be given.

1.10 Income context

- Additional context regarding the salary and compensation of the respondent.
- The respondent gave their own answer.
- This variable is not mandatory.
- Only 1 answer could be given.

1.11 Country

- The country in which the respondent works.
- The respondent gave their own answer.
- This variable is mandatory.
- Only 1 answer can be given.

1.12 State

- The state or states of the respondent in case the respondent works in the USA.
- The respondent chose from a premade list of answers.
- This variable is not mandatory.
- Multiple answers could be chosen.

1.13 City

- The city in which the respondent works.
- The respondent gave their own answer.
- This variable is mandatory.
- Only 1 answer can be given.

1.14 Overall professional years of eperience band

- The band in which the respondent has overall professional years of experience.
- The respondent chose from a premade list of answers.
- This variable is mandatory.
- Only 1 answer can be chosen.

1.15 Field professional years of eperience band

- The band in which the respondent professional years of experience in their current field.
- The respondent chose from a premade list of answers.
- This variable is mandatory.
- Only 1 answer can be chosen.

1.16 Education

- The respondent's highest level of education.
- The respondent chose from a premade list of answers.
- This variable is not mandatory.
- Only 1 answer could be chosen.

1.17 Gender

- The respondent's gender.
- The respondent chose from a premade list of answers (”Other/no answer” is an answer in the list).
- This variable is not mandatory.
- Only 1 answer could be chosen.

1.18 Race

- The respondent's race or races.
- The respondent chose from a premade list of answers (”Other/no answer” is an answer in the list).
- This variable is not mandatory.
- Multiple answers could be chosen.

2 Data cleansing and preparation

[214]:

```
# new column names - (CELL 5)
new_columns = [
    "datetime", "age_band", "industry", "job_title", "job_context", "salary",␣
  ↪"compensation", "currency", \
    "other_currency", "income_context", "country", "state", "city",␣
 ↪"overall_experience_band", "field_experience_band", \
    "education", "gender", "race"
]

# map old column names to new ones
mapping = {}
for i in range(0, len(data.columns)):
    mapping.update({data.columns[i]: new_columns[i]})

# rename columns
data = data.rename(mapping, axis='columns')

# check if columns are renamed
display(data.head())
```

```
             datetime age_band                       industry  \
0  4/27/2021 11:02:10    25-34   Education (Higher Education)
1  4/27/2021 11:02:22    25-34              Computing or Tech
2  4/27/2021 11:02:38    25-34  Accounting, Banking & Finance
3  4/27/2021 11:02:41    25-34                     Nonprofits
4  4/27/2021 11:02:42    25-34  Accounting, Banking & Finance

                                  job_title job_context  salary  compensation  \
0        Research and Instruction Librarian         NaN  55,000           0.0
1  Change & Internal Communications Manager         NaN  54,600        4000.0
2                      Marketing Specialist         NaN  34,000           NaN
3                           Program Manager         NaN  62,000        3000.0
4                        Accounting Manager         NaN  60,000        7000.0

  currency other_currency income_context         country           state  \
0      USD            NaN            NaN   United States   Massachusetts
1      GBP            NaN            NaN  United Kingdom             NaN
2      USD            NaN            NaN              US       Tennessee
3      USD            NaN            NaN             USA       Wisconsin
4      USD            NaN            NaN              US  South Carolina

          city overall_experience_band field_experience_band        education  \
0       Boston               5-7 years             5-7 years  Master's degree
1    Cambridge            8 - 10 years             5-7 years   College degree
2  Chattanooga             2 - 4 years           2 - 4 years   College degree
3    Milwaukee            8 - 10 years             5-7 years   College degree
4   Greenville            8 - 10 years             5-7 years   College degree

       gender   race
0       Woman  White
1  Non-binary  White
```

```
2       Woman  White
3       Woman  White
4       Woman  White
```

[215]:
```
# check what kind of dtype each column has - (CELL 6)
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27922 entries, 0 to 27921
Data columns (total 18 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   datetime               27922 non-null  object
 1   age_band               27922 non-null  object
 2   industry               27850 non-null  object
 3   job_title              27922 non-null  object
 4   job_context            7226 non-null   object
 5   salary                 27922 non-null  object
 6   compensation           20677 non-null  float64
 7   currency               27922 non-null  object
 8   other_currency         196 non-null    object
 9   income_context         3033 non-null   object
 10  country                27922 non-null  object
 11  state                  22945 non-null  object
 12  city                   27847 non-null  object
 13  overall_experience_band  27922 non-null  object
 14  field_experience_band  27922 non-null  object
 15  education              27711 non-null  object
 16  gender                 27757 non-null  object
 17  race                   27754 non-null  object
dtypes: float64(1), object(17)
memory usage: 3.8+ MB
None
```

[216]:
```
# change datetime format and dtype - (CELL 7)
data['datetime'] = pd.to_datetime(data['datetime'], infer_datetime_format=True)

# check if dtype and values are correct
print(data['datetime'].info())
display(data[['datetime']].head())
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 27922 entries, 0 to 27921
Series name: datetime
Non-Null Count  Dtype
--------------  -----
27922 non-null  datetime64[ns]
dtypes: datetime64[ns](1)
```

```
memory usage: 218.3 KB
None

             datetime
0 2021-04-27 11:02:10
1 2021-04-27 11:02:22
2 2021-04-27 11:02:38
3 2021-04-27 11:02:41
4 2021-04-27 11:02:42
```

[217]:

```
# change age band dtype - (CELL 8)
data['age_band'] = data['age_band'].astype('category')

# change industry dtype
data['industry'] = data['industry'].astype('category')

# check if dtype got changed
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27922 entries, 0 to 27921
Data columns (total 18 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   datetime                 27922 non-null  datetime64[ns]
 1   age_band                 27922 non-null  category
 2   industry                 27850 non-null  category
 3   job_title                27922 non-null  object
 4   job_context              7226 non-null   object
 5   salary                   27922 non-null  object
 6   compensation             20677 non-null  float64
 7   currency                 27922 non-null  object
 8   other_currency           196 non-null    object
 9   income_context           3033 non-null   object
 10  country                  27922 non-null  object
 11  state                    22945 non-null  object
 12  city                     27847 non-null  object
 13  overall_experience_band  27922 non-null  object
 14  field_experience_band    27922 non-null  object
 15  education                27711 non-null  object
 16  gender                   27757 non-null  object
 17  race                     27754 non-null  object
dtypes: category(2), datetime64[ns](1), float64(1), object(14)
memory usage: 3.5+ MB
None
```

[218]:

```
# check how many NaN industry has - (CELL 9)
print(data['industry'].isnull().sum())
```

```python
print(data.shape)

# only take rows with non-NaN
data = data[data['industry'].notna()]

# check if rows are removed
print(data['industry'].isnull().sum())
print(data.shape)
```

```
72
(27922, 18)
0
(27850, 18)
```

[219]:

```python
# https://towardsdatascience.com/string-matching-with-fuzzywuzzy-e982c61f8a84 -␣
 ↪(CELL 10)
# https://pypi.org/project/fuzzywuzzy/

# replace 'Health care' with 'Health Care'
data['industry'] = data['industry'].replace('Health care', 'Health Care')

# all answers that could be chosen for industry
premade_catgs = [
    'Accounting, Banking & Finance',
    'Agriculture or Forestry',
    'Art & Design',
    'Business or Consulting',
    'Computing or Tech',
    'Education (Primary/Secondary)',
    'Education (Higher Education)',
    'Engineering or Manufacturing',
    'Entertainment',
    'Government and Public Administration',
    'Health Care',
    'Hospitality & Events',
    'Insurance',
    'Law',
    'Law Enforcement & Security',
    'Leisure, Sport & Tourism',
    'Marketing, Advertising & PR',
    'Media & Digital',
    'Nonprofits',
    'Property or Construction',
    'Recruitment or HR',
    'Retail',
    'Sales',
    'Social Work',
    'Transport or Logistics',
    'Utilities & Telecommunications'
]

# adding this categories manually a good amount of answers involving it
premade_catgs.append('Library or Archiving')

# all answers present in 'industry'
all_catgs = list(data['industry'])

# get all text-input answers
input_catgs = []
for catg in all_catgs:
    if catg not in premade_catgs:
        input_catgs.append(catg)

# map text-input answers to a premade answer
catg_mapping = {}
catg_ratio_threshold = 75
for input_catg in input_catgs:
    matches = []
    for premade_catg in premade_catgs:

        # try to match every text-input answers to a premade answer when a␣
 ↪threshold is met
        TSR_score = fuzz.token_set_ratio(input_catg, premade_catg)
        PR_score = fuzz.partial_ratio(input_catg.lower(), premade_catg.lower())
        if TSR_score >= catg_ratio_threshold:
            matches.append(tuple([input_catg, premade_catg, TSR_score]))
        if PR_score >= catg_ratio_threshold:
            matches.append(tuple([input_catg, premade_catg, PR_score]))

    # if a text-input answer has multiple mactches, pick the one with the␣
 ↪highest score
    if len(matches) > 0:
        best_match = matches[0]
        for match in matches:
            if match[2] > best_match[2]:
                best_match = match
        catg_mapping.update({best_match[0]: best_match[1]})

    # if no match is found, text-input answer is mapped to 'Other'
    else:
        catg_mapping.update({input_catg: "Other"})
```

[220]:

```
# replace text-input answers for industry - (CELL 11)
data['industry'] = data['industry'].replace(catg_mapping)
```

```
# check categories present in industry
print(data['industry'].value_counts())
```

```
Computing or Tech                       4671
Education (Higher Education)            2465
Nonprofits                              2434
Health Care                             1899
Government and Public Administration    1897
Accounting, Banking & Finance           1798
Other                                   1794
Engineering or Manufacturing            1746
Marketing, Advertising & PR             1123
Law                                     1097
Business or Consulting                   861
Education (Primary/Secondary)            836
Media & Digital                          773
Insurance                                534
Retail                                   509
Recruitment or HR                        458
Property or Construction                 400
Utilities & Telecommunications           374
Art & Design                             365
Sales                                    353
Transport or Logistics                   316
Social Work                              273
Hospitality & Events                     266
Entertainment                            253
Agriculture or Forestry                  140
Leisure, Sport & Tourism                 100
Library or Archiving                      68
Law Enforcement & Security                47
Name: industry, dtype: int64
```

[221]:

```
# check how many unique job titles there are before lowercasing - (CELL 12)
print(len(data['job_title'].unique()))

# change job titles to lowercase
data['job_title'] = data['job_title'].str.lower()

# check how many unique job titles there are after lowercasing
print(len(data['job_title'].unique()))
```

```
14249
13005
```

[225]:

```python
# check how many NaN job context has - (CELL 13)
print(data['job_context'].isnull().sum())

# drop job context, too many NaN's
data = data.drop('job_context', axis='columns')

# check if job context got removed
print(data.shape)
```

```
20636
(27850, 17)
```

[226]:

```python
# remove american notation - (CELL 14)
data['salary'] = data['salary'].str.replace(',', '')

# change salary dtype
data['salary'] = data['salary'].astype(int)

# check if dtype got changed
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 27850 entries, 0 to 27921
Data columns (total 17 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   datetime                 27850 non-null  datetime64[ns]
 1   age_band                 27850 non-null  category
 2   industry                 27850 non-null  category
 3   job_title                27850 non-null  object
 4   salary                   27850 non-null  int64
 5   compensation             20629 non-null  float64
 6   currency                 27850 non-null  object
 7   other_currency           195 non-null    object
 8   income_context           3031 non-null   object
 9   country                  27850 non-null  object
 10  state                    22889 non-null  object
 11  city                     27775 non-null  object
 12  overall_experience_band  27850 non-null  object
 13  field_experience_band    27850 non-null  object
 14  education                27642 non-null  object
 15  gender                   27686 non-null  object
 16  race                     27683 non-null  object
dtypes: category(2), datetime64[ns](1), float64(1), int64(1), object(12)
memory usage: 3.5+ MB
None
```

[227]:

```python
# check which values are in other currency - (CELL 15)
print(data['other_currency'].value_counts())
```

```
USD
11
INR
10
NOK
10
SGD
9
MYR
8
DKK
8
AUD
7
BRL
6
PLN
5
CZK
4
NZD
4
NTD
2
ILS
2
GBP
2
KRW
2
CNY
2
MXN
2
None
2
ARS
2
Dkk
2
THB
2
IDR
1
```

```
PHP
1
RM
1
Polish Złoty
1
SAR
1
Philippine peso (PHP)
1
Australian Dollars
1
PhP (Philippine Peso)
1
Converted mine into USD for your easyness
1
Israeli Shekels
1
Many non-salary benefits - travel, free healthcare for self, very low for
family, non-taxable housing allowance      1
Equity
1
It’s marketed as £22000 but we get paid pro-rats, so no pay for the school
holidays.                                 1
additional compensation is for overtime (i am paid hourly) so it varies. i have
included an estimate                 1
Argentinian peso (ARS)
1
Rs
1
Argentine Peso
1
Philippine Pesos
1
Singapore Dollara
1
Rupees
1
5
1
PLN (Zwoty)
1
croatian kuna
1
up to 12% annual bonus
1
N/a
1
```

```
Canadian
1
na
1
47000
1
Thai Baht
1
Option to get 2x or 1.5x if taking on a weekend day in the summer
1
THAI  BAHT
1
Mexican Pesos
1
SGD
1
Euro
1
dkk
1
Korean Won
1
CAD
1
Danish Kroner
1
INR (Indian Rupee)
1
AUD Australian
1
Ils
1
LKR
1
IDR
1
China RMB
1
EUR
1
American Dollars
1
Additonal = Bonus plus stock
1
ZAR
1
RSU / equity
1
```

```
Bdt
1
Mexican pesos
1
BRL (R$)
1
Indian rupees
1
TTD
1
COP
1
canadian
1
DKK
1
Base plus Commission
1
SEK
1
BR$
1
Na
1
KWD
1
CHF
1
0
1
I work for an online state university, managing admissions data. Not direct tech
support.                             1
My bonus is based on performance up to 10% of salary
1
$76,302.34
1
Php
1
PLN (Polish zloty)
1
Overtime (about 5 hours a week) and bonus
1
czech crowns
1
6000 in stock grants annually
1
ILS (Shekel)
1
```

```
Nok
1
Sgd
1
Peso Argentino
1
Czk
1
KRW (Korean Won)
1
Philippine Peso
1
AUD and NZD aren't the same currency, and have absolutely nothing to do with
each other :(                             1
Taiwanese dollars
1
RMB (chinese yuan)
1
-
1
NIS (new Israeli shekel)
1
Canadian
1
US Dollar
1
AUD & NZD are not the same currency…
1
55,000
1
ILS/NIS
1
Norwegian kroner (NOK)
1
TRY
1
Stock
1
NGN
1
Name: other_currency, dtype: int64
```

[228]:

```
# check how many times 'Other' appears in currency - (CELL 16)
print(data['currency'].value_counts())
```

```
USD        23210
CAD         1660
GBP         1581
```

```
EUR          633
AUD/NZD      498
Other        154
CHF           37
SEK           37
JPY           23
ZAR           13
HKD            4
Name: currency, dtype: int64
```

[229]:

```
# drop currency with 'Other' value, too insignificant - (CELL 17)
print(data.shape)
data = data.drop(data[data['currency'] == 'Other'].index)

# drop other currency, too many NaN's
data = data.drop('other_currency', axis='columns')

# check if rows and column got removed
print(data.shape)
```

```
(27850, 17)
(27696, 16)
```

[230]:

```
# fill compensation with value '0' if NaN - (CELL 18)
data['compensation'] = data['compensation'].fillna(0)

# change compensation dtype
data['compensation'] = data['compensation'].astype(int)

# check if dtype changed
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 27696 entries, 0 to 27921
Data columns (total 16 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   datetime                  27696 non-null  datetime64[ns]
 1   age_band                  27696 non-null  category
 2   industry                  27696 non-null  category
 3   job_title                 27696 non-null  object
 4   salary                    27696 non-null  int64
 5   compensation              27696 non-null  int64
 6   currency                  27696 non-null  object
 7   income_context            3008 non-null   object
 8   country                   27696 non-null  object
 9   state                     22876 non-null  object
 10  city                      27621 non-null  object
 11  overall_experience_band  27696 non-null  object
 12  field_experience_band    27696 non-null  object
 13  education                27491 non-null  object
 14  gender                   27532 non-null  object
 15  race                     27532 non-null  object
dtypes: category(2), datetime64[ns](1), int64(2), object(11)
memory usage: 3.2+ MB
None
```

[231]:
```
# check how many NaN income context has - (CELL 19)
print(data['income_context'].isnull().sum())

# drop income context, too many NaN's
data = data.drop('income_context', axis='columns')

# check if job context got removed
print(data.shape)
```

```
24688
(27696, 15)
```

[232]:
```
# CELL 20
display(data.head())
```

```
             datetime age_band                        industry  \
0 2021-04-27 11:02:10    25-34   Education (Higher Education)
1 2021-04-27 11:02:22    25-34              Computing or Tech
2 2021-04-27 11:02:38    25-34  Accounting, Banking & Finance
3 2021-04-27 11:02:41    25-34                     Nonprofits
4 2021-04-27 11:02:42    25-34  Accounting, Banking & Finance

                                  job_title  salary  compensation currency  \
0        research and instruction librarian   55000             0      USD
1  change & internal communications manager   54600          4000      GBP
2                      marketing specialist   34000             0      USD
3                           program manager   62000          3000      USD
4                        accounting manager   60000          7000      USD

          country           state         city overall_experience_band  \
0   United States   Massachusetts       Boston               5-7 years
1  United Kingdom             NaN    Cambridge            8 - 10 years
2              US       Tennessee  Chattanooga             2 - 4 years
3             USA       Wisconsin    Milwaukee            8 - 10 years
4              US  South Carolina   Greenville            8 - 10 years

  field_experience_band        education      gender   race
0             5-7 years  Master's degree       Woman  White
1             5-7 years   College degree  Non-binary  White
```

```
2            2 - 4 years   College degree       Woman  White
3              5-7 years   College degree       Woman  White
4              5-7 years   College degree       Woman  White
```

[233]:
```
# change country to lowercase - (CELL 21)
data['country'] = data['country'].str.lower()

# remove punctuation
data['country'] = data['country'].str.replace('.', '')

# check which values are present in country
print(data['country'].value_counts())
```

```
united states
9275
usa
8568
us
3333
canada
1588
united states
672
uk
663
united kingdom
563
usa
485
united states of america
440
australia
313
germany
174
england
136
ireland
102
new zealand
99
canada
77
united kingdom
67
france
66
australia
```

```
66
united states of america
47
netherlands
46
spain
44
us
40
scotland
39
uk
37
sweden
34
belgium
33
england
32
switzerland
30
the netherlands
29
japan
27
america
21
new zealand
20
united state
19
germany
19
ireland
18
austria
17
finland
16
italy
14
unites states
13
south africa
13
netherlands
11
denmark
```

```
10
united stated
10
switzerland
7
israel
7
united sates
6
sweden
6
singapore
6
nz
6
india
6
the united states
6
scotland
6
england, uk
5
greece
5
u s
5
brazil
4
wales
4
united state of america
4
unitedstates
4
romania
4
portugal
4
latvia
4

4
china
4
pakistan
4
unites states
```

```
4
great britain
3
united statea
3
norway
3
mexico
3
isa
3
scotland, uk
3
south africa
3
is
3
thailand
3
hong kong
3
u s
3
puerto rico
3
unite states
2
the netherlands
2
remote
2
vietnam
2
canda
2
great britain
2
united stares
2
estonia
2
lithuania
2
slovenia
2
england, united kingdom
2
chile
```

```
2
uk (england)
2
northern ireland
2
usa tomorrow
2
philippines
2
poland
2
bulgaria
2
zimbabwe
2
ghana
2
kenya
2
cyprus
2
 us
2
the us
2
colombia
2
united sates of america
2
ua
2
bermuda
2
spain
2
japan
2
luxembourg
2
united status
2
united kingdom (england)
2
belgium
1
malaysia
1
uganda
```

```
1
england/uk
1
san francisco
1
united statws
1
sri lanka
1
ecuador
1
malta
1
us govt employee overseas, country withheld
1
usa-- virgin islands
1
contracts
1
morocco
1
africa
1
kuwait
1
currently finance
1
n/a (remote from wherever i want)
1
united stateds
1
united sttes
1
hungary
1
remote (philippines)
1
unites kingdom
1
global
1
nigeria
1
panamá
1
canada, ottawa, ontario
1
austria, but i work remotely for a dutch/british company
```

```
1
i was brought in on this salary to help with the ehr and very quickly was
promoted to current position but compensation was not altered
1
uniter statez
1
congo
1
uruguay
1
britain
1
usat
1
we don't get raises, we get quarterly bonuses, but they periodically asses
income in the area you work, so i got a raise because a 3rd party assessment
showed i was paid too little for the area we were located       1
luxemburg
1
northern ireland
1
norway
1
jamaica
1
usd
1
usa, but for foreign gov't
1
jordan
1
united statss
1
i work for a uae-based organization, though i am personally in the us
1
united  states
1
aotearoa new zealand
1
na
1
policy
1
us>
1
hong konh
1
united states is america
```

```
1
liechtenstein
1
company in germany i work from pakistan
1
canadá
1
united states of american
1
australian
1
uk, but for globally fully remote company
1
california
1
ukraine
1
unitef stated
1
wales, uk
1
united stares
1
croatia
1
england, united kingdom
1
usaa
1
south korea
1
united states- puerto rico
1
europe
1
from new zealand but on projects across apac
1
y
1
united y
1
mexico
1
wales (uk)
1
isle of man
1
northern ireland, united kingdom
```

```
1
qatar
1
uk, remote
1
unitied states
1
united states of americas
1
united arab emirates
1
rwanda
1
uk (northern ireland)
1
uk for us company
1
us of a
1
hong kong
1
canad
1
uniyes states
1
eritrea
1
uniyed states
1
cambodia
1
i am located in canada but i work for a company in the us
1
can
1
cayman islands
1
bangladesh
1
united statees
1
csnada
1
japan, us gov position
1
hartford
1
new zealand aotearoa
```

```
1
serbia
1
russia
1
uxz
1
united kindom
1
puerto rico
1
canada and usa
1
catalonia
1
$2,17584/year is deducted for benefits
1
france
1
italy (south)
1
jersey, channel islands
1
virginia
1
afghanistan
1
uss
1
uniteed states
1
united stattes
1
for the united states government, but posted overseas
1
usab
1
worldwide (based in us but short term trips aroudn the world)
1
englang
1
united statew
1
uae
1
canadw
1
bonus based on meeting yearly goals set w/ my supervisor
```

```
1
international
1
the bahamas
1
i earn commission on sales if i meet quota, i'm guaranteed another 16k min last
year i earned an additional 27k it's not uncommon for people in my space to earn
100k+ after commission                                 1
united statesp
1
costa rica
1
 united states
1
united statues
1
argentina
1
untied states
1
uniited states
1
united states of american
1
sierra leone
1
portugal
1
slovakia
1
nederland
1
united kingdomk
1
unted states
1
 new zealand
1
cuba
1
united states (i work from home and my clients are all over the us/canada/pr
1
australi
1
cote d'ivoire
1
from romania, but for an us based company
1
```

```
somalia
1
wales (united kingdom)
1
england, gb
1
danmark
1
uk (northern england)
1
malaysia
1
nl
1
bosnia and herzegovina
1
Name: country, dtype: int64
```

```
/tmp/ipykernel_41508/1744000476.py:5: FutureWarning: The default value of regex
will change from True to False in a future version. In addition, single
character regular expressions will *not* be treated as literal strings when
regex=True.
  data['country'] = data['country'].str.replace('.', '')
```

[234]:

```python
# list with correct values - (CELL 22)
correct_countries = [
    'united states of america',
    'united states',
    'usa',
    'united kingdom',
    'great britain'
    'uk',
    'england',
    'canada',
    'germany',
    'france',
    'spain',
    'scotland',
    'netherlands',
    'australia',
    'austria',
    'new zealand',
    'argentina',
    'italy',
    'finland',
    'wales',
    'ireland',
    'belgium',
    'switzerland',
    'japan',
    'south africa',
    'denmark'
]

input_countries = list(data['country'])

# map incorrect country to correct country
country_mapping = {}
country_ratio_threshold = 75
for input_country in input_countries:
    matches = []
    for correct_country in correct_countries:

        # try to match every text-input answers to a correct country variation␣
 ↪when a threshold is met
        TSR_score = fuzz.token_set_ratio(input_country, correct_country)
        if TSR_score >= country_ratio_threshold:
            matches.append(tuple([input_country, correct_country, TSR_score]))

    # if a text-input answer has multiple mactches, pick the one with the␣
 ↪highest score
    if len(matches) > 0:
        best_match = matches[0]
        for match in matches:
            if match[2] > best_match[2]:
                best_match = match
        country_mapping.update({best_match[0]: best_match[1]})

    # if no match is found, text-input answer is mapped to 'unknown'
    else:
        country_mapping.update({input_catg: "unknown"})
```

[235]:

```
# replace text-input answers for country - (CELL 23)
data['country'] = data['country'].replace(country_mapping)

# remove trailing whitespaces
data['country'] = data['country'].str.strip()

# check categories present in country
print(data['country'].value_counts())
```

```
usa
12439
united states of america
10464
```

```
canada
1673
uk
700
united kingdom
640
australia
381
germany
194
england
179
ireland
124
new zealand
123
united states
93
netherlands
89
france
67
scotland
48
spain
46
sweden
40
switzerland
37
belgium
34
japan
30
austria
18
south africa
17
finland
16
italy
15
denmark
11
u s
8
israel
7
```

```
singapore
6
wales
6
india
6
nz
6
greece
5
great britainuk
5
portugal
5
puerto rico
4
brazil
4
mexico
4
romania
4
norway
4
latvia
4
unitedstates
4
pakistan
4
china
4

4
hong kong
4
isa
3
thailand
3
is
3
slovenia
2
vietnam
2
remote
2
```

```
cyprus
2
malaysia
2
chile
2
philippines
2
poland
2
bulgaria
2
estonia
2
zimbabwe
2
ghana
2
kenya
2
the us
2
lithuania
2
bermuda
2
colombia
2
luxembourg
2
uk for us company
1
global
1
contracts
1
san francisco
1
we don't get raises, we get quarterly bonuses, but they periodically asses
income in the area you work, so i got a raise because a 3rd party assessment
showed i was paid too little for the area we were located        1
britain
1
ecuador
1
morocco
1
malta
```

```
1
worldwide (based in us but short term trips aroudn the world)
1
croatia
1
uganda
1
us govt employee overseas, country withheld
1
n/a (remote from wherever i want)
1
cayman islands
1
uruguay
1
luxemburg
1
south korea
1
hartford
1
ukraine
1
liechtenstein
1
hong konh
1
policy
1
na
1
i work for a uae-based organization, though i am personally in the us
1
jordan
1
kuwait
1
usd
1
jamaica
1
uk, but for globally fully remote company
1
california
1
europe
1
isle of man
```

```
1
sri lanka
1
y
1
congo
1
i was brought in on this salary to help with the ehr and very quickly was
promoted to current position but compensation was not altered
1
can
1
catalonia
1
serbia
1
russia
1
somalia
1
from romania, but for an us based company
1
cote d'ivoire
1
uxz
1
cuba
1
eritrea
1
panamá
1
$2,17584/year is deducted for benefits
1
jersey, channel islands
1
bangladesh
1
virginia
1
afghanistan
1
uss
1
cambodia
1
nl
1
```

```
slovakia
1
currently finance
1
sierra leone
1
nigeria
1
remote (philippines)
1
hungary
1
qatar
1
uk, remote
1
us of a
1
argentina
1
costa rica
1
i earn commission on sales if i meet quota, i'm guaranteed another 16k min last
year i earned an additional 27k it's not uncommon for people in my space to earn
100k+ after commission                                  1
rwanda
1
the bahamas
1
international
1
bonus based on meeting yearly goals set w/ my supervisor
1
united arab emirates
1
uae
1
bosnia and herzegovina
1
Name: country, dtype: int64
```

[236]:

```
# improve certain mappings - (CELL 24)
improved_country_mapping = {
    'usa': 'united states of america',
    'united states': 'united states of america',
    'uk': 'united kingdom',
    'england': 'united kingdom',
```

```python
    'great britain': 'united kingdom',
    'scotland': 'united kingdom',
    'wales': 'united kingdom'
}

# refine mappings for country
data['country'] = data['country'].replace(improved_country_mapping)

# check categories present in country
print(data['country'].value_counts())
```

```
united states of america
22996
canada
1673
united kingdom
1573
australia
381
germany
194
ireland
124
new zealand
123
netherlands
89
france
67
spain
46
sweden
40
switzerland
37
belgium
34
japan
30
austria
18
south africa
17
finland
16
italy
15
```

```
denmark
11
u s
8
israel
7
nz
6
singapore
6
india
6
greece
5
portugal
5
great britainuk
5
pakistan
4

4
unitedstates
4
china
4
latvia
4
romania
4
brazil
4
puerto rico
4
norway
4
mexico
4
hong kong
4
is
3
isa
3
thailand
3
philippines
2
```

```
zimbabwe
2
ghana
2
lithuania
2
estonia
2
bulgaria
2
remote
2
vietnam
2
malaysia
2
the us
2
poland
2
slovenia
2
chile
2
luxembourg
2
bermuda
2
kenya
2
cyprus
2
colombia
2
ecuador
1
i was brought in on this salary to help with the ehr and very quickly was
promoted to current position but compensation was not altered
1
panamá
1
morocco
1
uruguay
1
congo
1
uganda
```

```
1
malta
1
n/a (remote from wherever i want)
1
us govt employee overseas, country withheld
1
can
1
luxemburg
1
san francisco
1
jamaica
1
liechtenstein
1
hong konh
1
policy
1
na
1
i work for a uae-based organization, though i am personally in the us
1
jordan
1
usd
1
uk, but for globally fully remote company
1
remote (philippines)
1
california
1
europe
1
isle of man
1
y
1
ukraine
1
south korea
1
croatia
1
nigeria
```

```
1
costa rica
1
hungary
1
jersey, channel islands
1
cambodia
1
global
1
worldwide (based in us but short term trips aroudn the world)
1
uk for us company
1
hartford
1
uss
1
afghanistan
1
virginia
1
$2,17584/year is deducted for benefits
1
bangladesh
1
catalonia
1
uxz
1
russia
1
serbia
1
currently finance
1
united arab emirates
1
rwanda
1
us of a
1
britain
1
we don't get raises, we get quarterly bonuses, but they periodically asses
income in the area you work, so i got a raise because a 3rd party assessment
showed i was paid too little for the area we were located        1
```

```
qatar
1
sierra leone
1
uk, remote
1
argentina
1
cayman islands
1
i earn commission on sales if i meet quota, i'm guaranteed another 16k min last
year i earned an additional 27k it's not uncommon for people in my space to earn
100k+ after commission                                             1
the bahamas
1
international
1
bonus based on meeting yearly goals set w/ my supervisor
1
uae
1
kuwait
1
eritrea
1
slovakia
1
nl
1
sri lanka
1
somalia
1
from romania, but for an us based company
1
cote d'ivoire
1
cuba
1
contracts
1
bosnia and herzegovina
1
Name: country, dtype: int64
```

[237]:

```
# create df of correct countries - (CELL 25)
```

```
correct_countries_df = pd.DataFrame(correct_countries).rename({0: 'country'},␣
 ↪axis='columns')
display(correct_countries_df.head())
```

```
                   country
0  united states of america
1             united states
2                       usa
3            united kingdom
4           great britainuk
```

[238]:

```
# create filter to remove remaining countries - (CELL 26)
print(data.shape)
is_correct_country = data['country'].isin(correct_countries_df['country'])
data = data.drop(data[~is_correct_country].index)

# check if rows got dropped
print(data.shape)
```

```
(27696, 15)
(27450, 15)
```

[239]:

```
# change country dtype - (CELL 27)
data['country'] = data['country'].astype('category')

# check if dtype got changed
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 27450 entries, 0 to 27921
Data columns (total 15 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   datetime                 27450 non-null  datetime64[ns]
 1   age_band                 27450 non-null  category
 2   industry                 27450 non-null  category
 3   job_title                27450 non-null  object
 4   salary                   27450 non-null  int64
 5   compensation             27450 non-null  int64
 6   currency                 27450 non-null  object
 7   country                  27450 non-null  category
 8   state                    22831 non-null  object
 9   city                     27376 non-null  object
 10  overall_experience_band  27450 non-null  object
 11  field_experience_band    27450 non-null  object
 12  education                27248 non-null  object
 13  gender                   27289 non-null  object
 14  race                     27289 non-null  object
```

```
dtypes: category(3), datetime64[ns](1), int64(2), object(9)
memory usage: 2.8+ MB
None
```

[240]:

```
# create filter to drop 'united states of america' with no state - (CELL 28)
print(data.shape)
usa_no_state = (data['country'] == 'united states of america') & (data['state'].
 ↪isna())

# drop rows
data = data.drop(data[usa_no_state].index)

# check if rows got dropped
print(data.shape)
```

```
(27450, 15)
(27280, 15)
```

[241]:

```
# fill state NaN with value 'Not American' - (CELL 29)
data['state'] = data['state'].fillna('Not American')

# check if NaN still exists
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 27280 entries, 0 to 27921
Data columns (total 15 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   datetime                27280 non-null  datetime64[ns]
 1   age_band                27280 non-null  category
 2   industry                27280 non-null  category
 3   job_title               27280 non-null  object
 4   salary                  27280 non-null  int64
 5   compensation            27280 non-null  int64
 6   currency                27280 non-null  object
 7   country                 27280 non-null  category
 8   state                   27280 non-null  object
 9   city                    27210 non-null  object
 10  overall_experience_band 27280 non-null  object
 11  field_experience_band   27280 non-null  object
 12  education               27090 non-null  object
 13  gender                  27127 non-null  object
 14  race                    27132 non-null  object
dtypes: category(3), datetime64[ns](1), int64(2), object(9)
memory usage: 2.8+ MB
None
```

[242]:

```
# change country dtype to category - (CELL 30)
data['country'] = data['country'].astype('category')

# change state dtype to category
data['state'] = data['state'].astype('category')

# check if dtypes got changed
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 27280 entries, 0 to 27921
Data columns (total 15 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   datetime                 27280 non-null  datetime64[ns]
 1   age_band                 27280 non-null  category
 2   industry                 27280 non-null  category
 3   job_title                27280 non-null  object
 4   salary                   27280 non-null  int64
 5   compensation             27280 non-null  int64
 6   currency                 27280 non-null  object
 7   country                  27280 non-null  category
 8   state                    27280 non-null  category
 9   city                     27210 non-null  object
 10  overall_experience_band  27280 non-null  object
 11  field_experience_band    27280 non-null  object
 12  education                27090 non-null  object
 13  gender                   27127 non-null  object
 14  race                     27132 non-null  object
dtypes: category(4), datetime64[ns](1), int64(2), object(8)
memory usage: 2.6+ MB
None
```

[243]:

```
# drop city column (unable to clean without extensive effort) - (CELL 31)
print(data.shape)
data = data.drop('city', axis='columns')

# check if column got dropped
print(data.shape)
```

```
(27280, 15)
(27280, 14)
```

[244]:

```
# fix small typo in overall experience band and field experience band - (CELL␣
 ↪32)
data['overall_experience_band'] = data['overall_experience_band'].str.replace('␣
 ↪- ', '-')
```

```python
data['field_experience_band'] = data['field_experience_band'].str.replace(' - ', '-')

# check if typos got fixed
print(data['overall_experience_band'].value_counts(), '\n')
print(data['overall_experience_band'].value_counts())
```

```
11-20 years         9380
8-10 years          5264
5-7 years           4739
21-30 years         3547
2-4 years           2892
31-40 years          846
1 year or less       493
41 years or more     119
Name: overall_experience_band, dtype: int64

11-20 years         9380
8-10 years          5264
5-7 years           4739
21-30 years         3547
2-4 years           2892
31-40 years          846
1 year or less       493
41 years or more     119
Name: overall_experience_band, dtype: int64
```

[245]:

```python
# change overall experience band and field experience band to category - (CELL 33)
data['overall_experience_band'] = data['overall_experience_band'].astype('category')
data['field_experience_band'] = data['field_experience_band'].astype('category')

# check if dtypes got changed
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 27280 entries, 0 to 27921
Data columns (total 14 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   datetime                  27280 non-null  datetime64[ns]
 1   age_band                  27280 non-null  category
 2   industry                  27280 non-null  category
 3   job_title                 27280 non-null  object
 4   salary                    27280 non-null  int64
 5   compensation              27280 non-null  int64
 6   currency                 27280 non-null  object
 7   country                  27280 non-null  category
 8   state                    27280 non-null  category
 9   overall_experience_band  27280 non-null  category
 10  field_experience_band    27280 non-null  category
 11  education                27090 non-null  object
 12  gender                   27127 non-null  object
 13  race                     27132 non-null  object
dtypes: category(6), datetime64[ns](1), int64(2), object(5)
memory usage: 2.1+ MB
None
```

[249]:

```
# check values for education - (CELL 34)
print(data['education'].value_counts())
```

```
College degree                        13178
Master's degree                        8630
Some college                           1995
PhD                                    1383
Professional degree (MD, JD, etc.)     1295
High School                             609
Name: education, dtype: int64
```

[248]:

```
# check how many NaN's education has - (CELL 35)
print(data.shape)
print(data['education'].isnull().sum())

# drop rows with NaN's (no viable way to fill)
data = data[data['education'].notna()]

# check if rows got dropped
print(data.shape)
```

```
(27280, 14)
190
(27090, 14)
```

[250]:

```
# change education dtype - (CELL 36)
data['education'] = data['education'].astype('category')

# check if dtype has changed
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 27090 entries, 0 to 27921
Data columns (total 14 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
```

```
 0   datetime                 27090 non-null  datetime64[ns]
 1   age_band                 27090 non-null  category
 2   industry                 27090 non-null  category
 3   job_title                27090 non-null  object
 4   salary                   27090 non-null  int64
 5   compensation             27090 non-null  int64
 6   currency                 27090 non-null  object
 7   country                  27090 non-null  category
 8   state                    27090 non-null  category
 9   overall_experience_band  27090 non-null  category
 10  field_experience_band    27090 non-null  category
 11  education                27090 non-null  category
 12  gender                   26966 non-null  object
 13  race                     26974 non-null  object
dtypes: category(7), datetime64[ns](1), int64(2), object(4)
memory usage: 1.9+ MB
None
```

[252]:

```python
# check values for gender - (CELL 37)
print(data['gender'].value_counts())
```

```
Woman                            20732
Man                               5225
Non-binary                         734
Other or prefer not to answer      274
Prefer not to answer                 1
Name: gender, dtype: int64
```

[254]:

```python
# remap some gender values - (CELL 38)
gender_mapping = {
    'Other or prefer not to answer': 'No answer',
    'Prefer not to answer': 'No answer'
}

# replace values
data['gender'] = data['gender'].replace(gender_mapping)

# check if values got replaced
print(data['gender'].value_counts())
```

```
Woman          20732
Man             5225
Non-binary       734
No answer        275
Name: gender, dtype: int64
```

[257]:

```
# check for NaN's in gender - (CELL 39)
print(data['gender'].isnull().sum())

# fill NaN's with 'No answer'
data['gender'] = data['gender'].fillna('No answer')

# change gender type
data['gender'] = data['gender'].astype('category')

# check to see if NaN's are gone and dtype is changed
print(data.info())
```

```
0
<class 'pandas.core.frame.DataFrame'>
Int64Index: 27090 entries, 0 to 27921
Data columns (total 14 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   datetime                27090 non-null  datetime64[ns]
 1   age_band                27090 non-null  category
 2   industry                27090 non-null  category
 3   job_title               27090 non-null  object
 4   salary                  27090 non-null  int64
 5   compensation            27090 non-null  int64
 6   currency                27090 non-null  object
 7   country                 27090 non-null  category
 8   state                   27090 non-null  category
 9   overall_experience_band  27090 non-null  category
 10  field_experience_band   27090 non-null  category
 11  education               27090 non-null  category
 12  gender                  27090 non-null  category
 13  race                    26974 non-null  object
dtypes: category(8), datetime64[ns](1), int64(2), object(3)
memory usage: 1.7+ MB
None
```

[261]:

```
# drop race (prefered to not use this data for predictive modeling) - (CELL 40)
data = data.drop('race', axis='columns')

# check if column got dropped
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 27090 entries, 0 to 27921
Data columns (total 13 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   datetime                27090 non-null  datetime64[ns]
 1   age_band                 27090 non-null  category
 2   industry                 27090 non-null  category
 3   job_title                27090 non-null  object
 4   salary                   27090 non-null  int64
 5   compensation             27090 non-null  int64
 6   currency                 27090 non-null  object
 7   country                  27090 non-null  category
 8   state                    27090 non-null  category
 9   overall_experience_band  27090 non-null  category
 10  field_experience_band    27090 non-null  category
 11  education                27090 non-null  category
 12  gender                   27090 non-null  category
dtypes: category(8), datetime64[ns](1), int64(2), object(2)
memory usage: 1.5+ MB
None
```

[266]:

```python
# save cleaned data - (CELL 41)
data.to_csv('Surveys_cleaned.csv', index=False)
```