Data Wrangling Lab

Estimated time needed: **45 to 60** minutes

In this assignment you will be performing data wrangling.

Objectives

In this lab you will perform the following:

- Identify duplicate values in the dataset.
- Remove duplicate values from the dataset.
- Identify missing values in the dataset.
- Impute the missing values in the dataset.
- Normalize data in the dataset.

Hands on Lab

Import pandas module.

In [1]:
```
import pandas as pd
```

Load the dataset into a dataframe.

In [2]:
```
df = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DA0321EN-SkillsNetwork/LargeData/m1_
```

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11552 entries, 0 to 11551
Data columns (total 85 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Respondent              11552 non-null  int64  
 1   MainBranch              11552 non-null  object 
 2   Hobbyist                11552 non-null  object 
 3   OpenSourcer             11552 non-null  object 
 4   OpenSource              11471 non-null  object 
 5   Employment              11552 non-null  object 
 6   Country                 11552 non-null  object 
 7   Student                 11499 non-null  object 
 8   EdLevel                 11436 non-null  object 
 9   UndergradMajor          10812 non-null  object 
 10  EduOther                11388 non-null  object 
 11  OrgSize                 11454 non-null  object 
 12  DevType                 11485 non-null  object 
 13  YearsCode               11543 non-null  object 
 14  Age1stCode              11539 non-null  object 
 15  YearsCodePro            11536 non-null  object 
 16  CareerSat               11552 non-null  object 
 17  JobSat                  11551 non-null  object 
 18  MgrIdiot                11054 non-null  object 
 19  MgrMoney                11050 non-null  object 
 20  MgrWant                 11054 non-null  object 
 21  JobSeek                 11552 non-null  object 
 22  LastHireDate            11552 non-null  object 
 23  LastInt                 11129 non-null  object 
 24  FizzBuzz                11515 non-null  object 
 25  JobFactors              11549 non-null  object 
 26  ResumeUpdate            11511 non-null  object 
 27  CurrencySymbol          11552 non-null  object 
 28  CurrencyDesc            11552 non-null  object 
 29  CompTotal               10737 non-null  float64
 30  CompFreq                11346 non-null  object 
 31  ConvertedComp           10730 non-null  float64
 32  WorkWeekHrs             11427 non-null  float64
 33  WorkPlan                11429 non-null  object 
 34  WorkChallenge           11384 non-null  object 
 35  WorkRemote              11544 non-null  object 
 36  WorkLoc                 11520 non-null  object 
 37  ImpSyn                  11547 non-null  object 
 38  CodeRev                 11551 non-null  object 
 39  CodeRevHrs              9083 non-null   float64
 40  UnitTests               11523 non-null  object 
 41  PurchaseHow             11354 non-null  object 
 42  PurchaseWhat            11514 non-null  object 
 43  LanguageWorkedWith      11541 non-null  object 
 44  LanguageDesireNextYear  11415 non-null  object 
 45  DatabaseWorkedWith      11096 non-null  object 
 46  DatabaseDesireNextYear  10497 non-null  object 
 47  PlatformWorkedWith      11130 non-null  object 
 48  PlatformDesireNextYear  10991 non-null  object 
 49  WebFrameWorkedWith      10139 non-null  object 
 50  WebFrameDesireNextYear  9918 non-null   object 
 51  MiscTechWorkedWith      9343 non-null   object 
 52  MiscTechDesireNextYear  10078 non-null  object 
 53  DevEnviron              11523 non-null  object 
 54  OpSys                   11518 non-null  object 
 55  Containers              11470 non-null  object 
 56  BlockchainOrg           9198 non-null   object 
 57  BlockchainIs            8915 non-null   object 
 58  BetterLife              11452 non-null  object 
 59  ITperson                11517 non-null  object 
 60  OffOn                   11514 non-null  object 
 61  SocialMedia             11251 non-null  object 
 62  Extraversion            11532 non-null  object 
 63  ScreenName              11039 non-null  object 
 64  SOVisit1st              11227 non-null  object 
 65  SOVisitFreq             11547 non-null  object 
 66  SOVisitTo               11551 non-null  object 
 67  SOFindAnswer            11549 non-null  object 
 68  SOTimeSaved             11501 non-null  object 
 69  SOHowMuchTime           9616 non-null   object 
 70  SOAccount               11551 non-null  object 
 71  SOPartFreq              10404 non-null  object 
 72  SOJobs                  11546 non-null  object 
 73  EntTeams                11547 non-null  object 
 74  SOComm                  11552 non-null  object 
 75  WelcomeChange           11463 non-null  object 
 76  SONewContent            9557 non-null   object 
 77  Age                     11255 non-null  float64
 78  Gender                  11477 non-null  object 
 79  Trans                   11429 non-null  object 
 80  Sexuality               11005 non-null  object 
 81  Ethnicity              10869 non-null  object
 82  Dependents             11408 non-null  object
 83  SurveyLength           11533 non-null  object
 84  SurveyEase             11538 non-null  object
dtypes: float64(5), int64(1), object(79)
memory usage: 7.5+ MB
```

Finding duplicates

In this section you will identify duplicate values in the dataset.

Find how many duplicate rows exist in the dataframe.

In [4]:
```
duplicate_rows = df[df.duplicated()]
len(duplicate_rows)
```

Out[4]:
```
154
```

In [5]:
```
#How many duplicate values are there in the column Respondent?

len(df['Respondent'])-len(df['Respondent'].drop_duplicates())
```

Out[5]:
```
154
```

Removing duplicates

Remove the duplicate rows from the dataframe.

In [6]:
```
df.drop_duplicates(keep='first', inplace=True)
```

Verify if duplicates were actually dropped.

In [7]:
```
duplicate_rows = df[df.duplicated()]
len(duplicate_rows)
```

Out[7]:
```
0
```

In [8]:
```
#After removing the duplicate rows, how many rows are there in the dataset?
len(df)
```

Out[8]:
```
11398
```

In [9]:
```
#After removing the duplicate rows, how many unique rows are there in the column Respondent?
len(df['Respondent'])
```

Out[9]:
```
11398
```

Finding Missing values

Find the missing values for all columns.

In [10]:
```
missing_data = df.isnull() #"True" means the value is missing
missing_data.head(5)
```

Out[10]:

	Respondent	MainBranch	Hobbyist	OpenSourcer	OpenSource	Employment	Country	Student	EdLevel	UndergradMajor	...	WelcomeCha
0	False	False	False	False	False	False	False	False	False	False	...	F
1	False	False	False	False	False	False	False	False	False	False	...	F
2	False	False	False	False	False	False	False	False	False	False	...	F
3	False	False	False	False	False	False	False	False	False	True	...	F
4	False	False	False	False	False	False	False	False	False	False	...	F

5 rows × 85 columns

Find out how many rows are missing in the column 'EdLevel' and 'Country'

In [11]:
```
missing_data['EdLevel'].value_counts()
```

Out[11]:
```
False    11286
True       112
Name: EdLevel, dtype: int64
```

In [12]:
```
missing_data['Country'].value_counts()
```

```
Out[12]: False    11398
         Name: Country, dtype: int64
```

Imputing missing values

Find the value counts for the column WorkLoc.

```
In [13]: df['WorkLoc'].value_counts()
```

```
Out[13]: Office                                          6806
         Home                                            3589
         Other place, such as a coworking space or cafe   971
         Name: WorkLoc, dtype: int64
```

Identify the value that is most frequent (majority) in the WorkLoc column.

```
In [14]: df['WorkLoc'].value_counts().idxmax()
```

```
Out[14]: 'Office'
```

```
In [15]: #What is the majority category under the column Employment?
         df['Employment'].value_counts().idxmax()
```

```
Out[15]: 'Employed full-time'
```

```
In [16]: #Under the column " UndergradMajor", which category has the minimum number of rows?
         df['UndergradMajor'].value_counts().idxmin()
```

```
Out[16]: 'A health science (ex. nursing, pharmacy, radiology)'
```

Impute (replace) all the empty rows in the column WorkLoc with the value that you have identified as majority.

```
In [17]: import numpy as np
         df['WorkLoc'].replace(np.nan, 'Office', inplace=True)
```

After imputation there should ideally not be any empty rows in the WorkLoc column.

Verify if imputing was successful.

```
In [18]: missing_data = df.isnull()
         len(df['WorkLoc']) ==len(missing_data[missing_data['WorkLoc'] == False])
```

```
Out[18]: True
```

Normalizing data

There are two columns in the dataset that talk about compensation.

One is "CompFreq". This column shows how often a developer is paid (Yearly, Monthly, Weekly).

The other is "CompTotal". This column talks about how much the developer is paid per Year, Month, or Week depending upon his/her "CompFreq".

This makes it difficult to compare the total compensation of the developers.

In this section you will create a new column called 'NormalizedAnnualCompensation' which contains the 'Annual Compensation' irrespective of the 'CompFreq'.

Once this column is ready, it makes comparison of salaries easy.

List out the various categories in the column 'CompFreq'

```
In [19]: df['CompFreq'].value_counts()
```

```
Out[19]: Yearly     6073
         Monthly    4788
         Weekly      331
         Name: CompFreq, dtype: int64
```

```
In [20]: #delete the rows in which CompFreq or CompTotal is missing
         df.dropna(subset=['CompFreq', 'CompTotal'], axis=0, inplace=True)
```

Create a new column named 'NormalizedAnnualCompensation'. Use the hint given below if needed.

Double click to see the **Hint**.

In [21]:
```
#create a new empty column named 'NormalizedAnnualCompensation'
df['NormalizedAnnualCompensation'] = ""

#If CompFreq is Yearly, use the exising value in CompTotal
#If CompFreq is Monthly, multiply the value in CompTotal by 12
#If  CompFreq is Weekly,  multiply the value in CompTotal by 52 (weeks in an year)
conditions = [(df['CompFreq'] == 'Yearly'), (df['CompFreq'] == 'Monthly'), (df['CompFreq'] == 'Weekly')]
choices = [df['CompTotal'], df['CompTotal']*12, df['CompTotal']*52]
df['NormalizedAnnualCompensation'] = np.select(conditions, choices, default=0)

df[['CompFreq', 'CompTotal', 'NormalizedAnnualCompensation']].head(15)
```

Out[21]:

	CompFreq	CompTotal	NormalizedAnnualCompensation
0	Yearly	61000.0	61000.0
1	Yearly	138000.0	138000.0
2	Yearly	90000.0	90000.0
3	Monthly	29000.0	348000.0
4	Yearly	90000.0	90000.0
5	Monthly	9500.0	114000.0
6	Monthly	3000.0	36000.0
7	Yearly	103000.0	103000.0
8	Yearly	69000.0	69000.0
9	Monthly	8000.0	96000.0
10	Monthly	7000.0	84000.0
11	Yearly	114000.0	114000.0
12	Weekly	2000.0	104000.0
13	Weekly	22000.0	1144000.0
14	Monthly	96000.0	1152000.0

In [22]:
```
#What is the median NormalizedAnnualCompensation?
df['NormalizedAnnualCompensation'].median()
```

Out[22]:
```
100000.0
```

Authors

Ramesh Sannareddy

Other Contributors

Rav Ahuja

Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2020-10-17	0.1	Ramesh Sannareddy	Created initial version of the lab