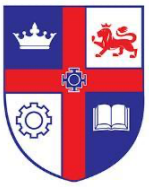
LONDON METROPOLITAN UNIVERSITY



Islington college
(इस्लिङ्टन कलेज)

CC5067NI-Smart Data Discovery

60% Individual Coursework

2023-24 Spring

Student Name: Sujal Nakarmi

London Met ID: 22067808

College ID: np01cp4a220192

Assignment Due Date: Monday, May 13, 2024

Assignment Submission Date: Monday, May 13, 2024

Word Count: 4312

I confirm that I understand my coursework needs to be submitted online via MySecondTeacher under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.

Table of Contents

1. Introduction *1*

2. Data Understanding *2*

a. To understand what your data resources are and the characteristics of those resources. Write down your findings 2

3. Data Preparation *7*

a. Write a python program to load data into pandas DataFrame. 7

b. Write a python program to remove unnecessary columns i.e., salary and salary currency 11

c. Write a python program to remove the NaN missing values from updated dataframe. 12

d. Write a python program to check duplicates value in the dataframe. 14

e. Write a python program to see the unique values from all the columns in the dataframe. 17

f. Rename the experience level columns as below. 18

4. Data Analysis *20*

a. Write a Python program to show summary statistics of sum, mean, standard deviation, skewness, and kurtosis of any chosen variable. 20

b. Write a Python program to calculate and show correlation of all variables. 21

5. Data Exploration *22*

a. Write a python program to find out top 15 jobs. Make a bar graph as well. 22

b. Which job has the highest salaries? Illustrate with bar graph. 24

c. Write a python program to find out salaries based on experience level. Illustrate it through bar graph. 27

d. Write a Python program to show histogram and box plot of any chosen different variables. Use proper labels in the graph. 29

Conclusion *32*

References *33*

Table of Figures

Figure 1 Importing libraries 7
Figure 2 Uploading csv file 8
Figure 3 Reading csv file 9
Figure 4 Data inconsistency (Data Scientist and Machine Learning) 9
Figure 5 Data inconsistency(Data Analyst and Data Engineer) 10
Figure 6 Data inconsistency (Data Architect, Data Analytics and Data Science) 10
Figure 7 Dropping Columns 11
Figure 8 Checking null value 12
Figure 9 Checking null value count in each column 13
Figure 10 Dropping null value 13
Figure 11 Checking duplicate value 14
Figure 12 Checking duplicated value total number 14
Figure 13 Using loc method to retrive rows and column 15
Figure 14 Dropping duplicate value 16
Figure 15 Using for each loop to retieve unique value from all column 17
Figure 16 Replacing value in column 18
Figure 17 Showing output after being replaced 19
Figure 18 Summary stats 20
Figure 19 Summary stats for work year column 21
Figure 20 Result of stats 21
Figure 21 Showing Correlation 22
Figure 22 Displaying top 15 jobs 22
Figure 23 Bar Graph of top 15 jobs 23
Figure 24 Bar Graph of top 15 jobs 23
Figure 25 Showing highest salaries of job title 24
Figure 26 Output of highest salaries of job 25
Figure 27 Code for bar graph of highest salaries job 25
Figure 28 Bar graph of highest salaries job 26

Figure 29 Retrieving max salary based on experience level 27
Figure 30 Code for Bar Graph of max salary based on experience level 28
Figure 31 Bar Graph of max salary based on experience level 28
Figure 32 Code for histogram of work year and salary in usd column 29
Figure 33 Histogram of work year and salary in usd column 30
Figure 34 Code for Boxplot 30
Figure 35 Box plot of salary in usd column 30
Figure 36 code for box plot of work year column 31
Figure 37 Box plot of work year column 31

Table of Tables

Table 1 Work Year Column 3
Table 2 Experience Level Column 3
Table 3 Employment Type Column 4
Table 4 Job Title column 4
Table 5 Salary column 4
Table 6 Salary Currency Column 5
Table 7 Salary in usd column 5
Table 8 Employment Residence column 5
Table 9 Remote Ratio Column 6
Table 10 Company location column 6
Table 11 Company size column 6

1. Introduction

There is a data set provided to us where we have to convert the data set into Pandas structure DataFrame for data manipulation. The data set represents various job title along with their experience level and the salary the earn in different current provided by the company which differs from the location of the company located and also can differ from the size of the company.

The data manipulation part is divided into 4 parts which are data understanding, data preparation, data analysis and data exploration. The data understanding part is about describing or explaining what the data set is about and what each column holds like which data type. In data preparation the csv file is read and loaded as dataframe, there is replacement of the data inconsistency to make it consistent, removing the null value or NaN value, displaying unique value from all the column, displaying the duplicating value and removing all the duplicated value, in data analysis summary statistics are shown of the numeric column along with the correlation, in data exploration there are representation of the graph, histogram and box plot which are the function of matplotlib, This also can help in data analysis through visual representation.

In documentation all Python programs have screen shots of testing, results, and brief user guide, along with adequate comments in the code.

2. Data Understanding

a. To understand what your data resources are and the characteristics of those resources. Write down your findings.

The data set is about. There are altogether 3755 rows and 11 different columns in the data set which are work_year, experience_level, employment_type, job_title, salary, salary_currency, salary_in_usd, employee_residence, remote_ratio, company_location, company_size. Each column represents different value and have different data type.

The work year column describes which year the data is recorded, experience level describes the level of experience an individual has in the company, employment type represent the what type of employment does the individual has and salary is the money an individual earn base on the salary currency. The salary_in_usd column represent the value of salary converted into usd, employee-residence defines the place where the individual nationality is from, remote_ratio describes company_location represents the location of the company in which country it is located and lastly company size represents the size of the company.

It represents salary variations as the salary is different or based upon different job titles, experience level and company sizes. Different location of company has different salary rate for the employee according to the job title and experience level. The salary also differs according to the company size. Each company located in different location provide different salary currency accordingly to their respective country which might lead to differ in the rate of salary. It shows the remote work opportunity in different fields. It allows for comparison which might be helpful for choosing career.

Table 1 Work Year Column

S. No	Column Name	Description	Data Type
1.	work_year	In this column there are 4 unique value 2020, 2021, 2022 and 2023 which are all integers. It means that all the other columns value are based or recorded on these years.	Integer

Table 2 Experience Level Column

S. No	Column Name	Description	Data Type
2.	experience_level	In this column also there are 4 unique value SE, EN, MI and EX and have there occurrence in the DataFrame accordingly. SE means Senior Level/Expert. EN means Entry Level, MI means Medium Level/Intermediate, and EX means Executive Level. These are the experience level of an individual which differs accordingly to all the other columns. The executive level is the highest level of experience with leadership skills and most experience, then comes Senior Level who have deep knowledge in the area, then Middle Level with some experience and at last entry level with no experience they are just the starters.	String

Table 3 Employment Type Column

S. No	Column Name	Description	Data Type
3	employment_type	The employment_type column also consists of 4 unique value which are FT, CT, FL, PT. FT stands for Full time, CT stands for Contract time, FL stands for and PT stands for Part Time. FL stands for free lancing Full time typically refers to job where an individual have to work 35-40 hours per week, Contract means the individual is not permanently assigned they are working for the company up to certain date, Part time means an individual is working in an company for only some hour as it is not their main job. Free lancing is not a long term commitment they are just working for projects.	String

Table 4 Job Title column

S. No	Column Name	Description	Data Type
4	job_title	This column holds job titles of an individual which describes which field is the individual is from and what kind of job is the individual has. For example Software Engineer, Data Scientist, Data Analyst, Manager, Staff Data science , Head of Data, Cloud Engineer and so on.	String

Table 5 Salary column

S. No	Column Name	Description	Data Type
5	salary	The salary column holds integer value. It represents how much an individual earn based on the experience level and job title. The salary depends on the salary currency and company locations although it is not explicitly defined.	Integer

Table 6 Salary Currency Column

S. No	Column Name	Description	Data Type
6	salary_currency	The salary currency column represents in which currency is the company paying the salary to their employees. The salary currency can vary from country to country some might have high rate, and some might have low rate. The currency is different according to the country like USD is the currency for United States, EUR is the currency code for Euro and is used in eurozone countries which is Europe and so on.	String

Table 7 Salary in usd column

S. No	Column Name	Description	Data Type
7	salary_in_usd	This column also holds integer data type but the all the salary from the salary column is converted into USD currency despite the company location. Converting this makes it easier to compare the salary based on experience level, job titles all over the world.	Integer

Table 8 Employment Residence column

S. No	Column Name	Description	Data Type
8	employee_residence	This column holds string value. It defines the location of the employees which means where originally the employee is from. For example, US means the Employee is from United States, NZ is New Zealand, CA is Canada, AU is Australia, BR is Brazil and so on.	String

Table 9 Remote Ratio Column

S. No	Column Name	Description	Data Type
9	remote_ratio	The remote ratio column represents the percentage of the remote work allowed for the specific job title. Here for example for Machine Learning the remote ratio Is 100 which means the employee can work entirely from work from home, and when the remote ratio is 0 it means that only physical presence in office is allowed for work.	Integer

Table 10 Company location column

S. No	Column Name	Description	Data Type
10	company_location	This column holds the location where the company is located. It helps us to know information about the company like in which country is the company located at.	String

Table 11 Company size column

S. No	Column Name	Description	Data Type
11	company_size	This column holds the String values. There are altogether 3 values in this column which are L, M and S. L stands for Large size company, M stands for Medium size company and S stands for Small size company. Depending upon the company size the salary may differ.	String

3. Data Preparation

a. Write a python program to load data into pandas DataFrame.

Panda is a powerful open-source python library for data manipulation and analysis. It provides data structures and data analysis tools for working with structured (tabular, single dimensional) and time series data. (Geeks, 2024)

For creating static, animated, and interactive visualizations in Python Matplotlib is a comprehensive library. To produce high-quality plots, it provides wide range of tools and command. (matoplotlib, 2023)

The first step is to import pandas and matplotlib for using these libraries in the coursework. The code for importing these libraries is shown below where import is the keyword pandas is the libraries and pd is an alias where pandas is assigned, alias is short name, usually given to a function, module in programming language for making the code short and easy to read and understand. The name of alias can be anything of our choice. Same goes for "**matpotlib.pyplot**". The alias is given plt so that we don't have to write matpotlib.pyolot everytime we need to use the library method. It is time consuming and the code extends.

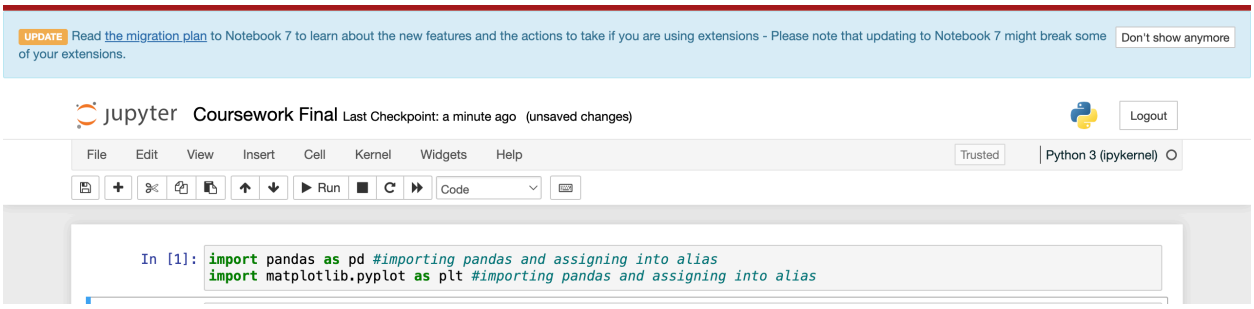


Figure 1 Importing libraries

Now the dataset is loaded in the same folder where we will save and run the code and manipulate the dataset. We add the dataset by clicking in the upload button and choose from download.

Figure 2 Uploading csv file

To load the data into pandas DataFrame we use the function called read_csv(). Dataframe is the data structure provided by the panda's library. It is like spreadsheet and SQL file which is a two-dimensional data structure, consisting of rows and column where each column can have a different data type.

"**read_csv()"** is a function used to import file with CSV extension to a DataFrame format. Here I have used this function to import CSV file name "DataScience.csv" and stored in a variable called Data. Then data is written to print the DataFrame. Pd is the alias we created when importing Panda's libraries.

Write a python program to load data into pandas DataFrame

```
In [2]: data = pd.read_csv('DataScienceSalaries_.csv')

        #read.csv() is a function of dataframe to import DataScienceSalaries.csv and convert into DataFrame and storing in d

In [3]: data.head() #printing by using head function to display the first 5 rows to check if the file is being imported or n
```

Out[3]:

	work_year	experience_level	employment_type	job_title	salary	salary_currency	salary_in_usd	employee_residence	remote_ratio	company_location	com
0	2023	SE	FT	Principal Data Scientist	80000	EUR	85847	ES	100	ES	
1	2023	MI	CT	ML Engineer	30000	USD	30000	US	100	US	
2	2023	MI	CT	ML Engineer	25500	USD	25500	US	100	US	
3	2023	SE	FT	Data Scientist	175000	USD	175000	CA	100	CA	
4	2023	SE	FT	Data Scientist	120000	USD	120000	CA	100	CA	

```
In [ ]:
```

Figure 3 Reading csv file

Removing data inconsistency is one of the major thing while performing data manipulation and execution. To remove data inconsistency from column name job_title we can use replace function. The syntax follows a dictionary-like structure, where the original values are specified as keys and the replacement values are provided as corresponding values. The default value is inplace = false which means it will only take effect temporarily only that specific code is executed but to make the effect directly to the original DataFrame permanently we must mention true in the parameter.

Replacing all Data which are inconsistent

```
In [4]: data['job_title'].replace({'Principal Data Scientist':'Data Scientist',
                                   'Product Data Scientist':'Data Scientist',
                                   'Staff Data Scientist':'Data Scientist',
                                   'Applied Data Scientist':'Data Scientist',
                                   'Data Scientist Lead':'Data Scientist',
                                   'Lead Data Scientist':'Data Scientist'} ,inplace = True)

        #In column job_title replacing data which are inconsistent. This is the code for replacing all heading
        #using replace function relating to data scientist and replacing them as Data Scientist

In [5]: data['job_title'].replace({'ML Engineer':'Machine Learning ',
                                   'Applied Machine Learning Engineer':'Machine Learning ',
                                   'Machine Learning Engineer' : 'Machine Learning ',
                                   'Machine Learning Researcher':'Machine Learning ',
                                   'Machine Learning Scientist':'Machine Learning ',
                                   'Machine Learning Infrastructure Engineer':'Machine Learning ',
                                   'Head of Machine Learning':'Machine Learning ',
                                   'Applied Machine Learning Scientist':'Machine Learning ',
                                   'Machine Learning Manager': 'Machine Learning ',
                                   'Lead Machine Learning Engineer': 'Machine Learning ',
                                   'Principal Machine Learning Engineer': 'Machine Learning ',
                                   'Machine Learning Research Engineer': 'Machine Learning ',
                                   'Machine Learning Software Engineer': 'Machine Learning ',
                                   'Machine Learning Developer':'Machine Learning '} ,inplace = True)
```

Figure 4 Data inconsistency (Data Scientist and Machine Learning)

```
In [6]: data['job_title'].replace({
                                'Lead Data Analyst':'Data Analyst',
                                'Financial Data Analyst':'Data Analyst',
                                'Product Data Analyst':'Data Analyst',
                                'Marketing Data Analyst':'Data Analyst',
                                'Principal Data Analyst':'Data Analyst',
                                'Finance Data Analyst':'Data Analyst',
                                'Data Quality Analyst':'Data Analyst',
                                'Compliance Data Analyst':'Data Analyst',
                                'Business Data Analyst':'Data Analyst',
                                'Staff Data Analyst':'Data Analyst',
                                'BI Data Analyst':'Data Analyst'} ,inplace = True)

        #In column job_title replacing data which are inconsistent. This is the code for replacing all heading using replace
        #function relating to Data Analyst and replacing them as Data Analyst

In [7]: data['job_title'].replace({

                                'Big Data Engineer':'Data Engineer',
                                'Software Data Engineer':'Data Engineer',
                                'Cloud Data Engineer':'Data Engineer',
                                'Azure Data Engineer':'Data Engineer',
                                'Marketing Data Engineer':'Data Engineer',
                                'Lead Data Engineer':'Data Engineer',
                                'BI Data Engineer':'Data Engineer',
                                'Principal Data Engineer':'Data Engineer'} ,inplace = True)
```

Figure 5 Data inconsistency(Data Analyst and Data Engineer)

```
In [8]: data['job_title'].replace({
                                'Big Data Architect':'Data Architect',
                                'Principal Data Architect':'Data Architect',
                                'Cloud Data Architect':'Data Architect' } ,inplace = True)


        #In column job_title replacing data which are inconsistent. This is the code for replacing all heading
        #using replace function relating to Data Architect and replacing them as Data Architect'''

In [9]: data['job_title'].replace ({'Data Analytics Manager':'Data Analytics',
                                'Data Analytics Engineer':'Data Analytics',
                                'Data Analytics Consultant':'Data Analytics',
                                'Data Analytics Specialist':'Data Analytics',
                                'Data Analytics Lead':'Data Analytics',} ,inplace= True)

        #In column job_title replacing data which are inconsistent. This is the code for replacing all heading
        #using replace function relating to Data Analytics and replacing them as Data Analytics

In [10]: data['job_title'].replace({'Data Science Manager':'Data Science',
        'Director of Data Science':'Data Science',
        'Data Science Lead':'Data Science',
        'Data Science Consultant':'Data Science',
        'Head of Data Science':'Data Science',
        'Data Science Engineer':'Data Science',
        'Data Science Tech Lead':'Data Science'},inplace= True)
```

Figure 6 Data inconsistency (Data Architect, Data Analytics and Data Science)

b. Write a python program to remove unnecessary columns i.e., **salary and salary currency.**

To remove anything from DataFrame we use drop() function. Data is the Variable where DataFrame is stored. Here drop function is used to drop two columns from the DataFrame salary and salary currency. We write the name of the columns inside the drop function parameter with big bracket and with a single inverted comma, and axis = 1 is used to indicate the column, inplace = True is written because in the absence of inplace true the columns will be only removed for once(temporarily), when we run that particular code. The default value is always inplace = False. But in the presence of inplace = True, the columns will be deleted permanently affecting underlying data. Therefore, in the output the two columns are removed after printing data.

```
In [11]: data.drop(['salary','salary_currency'], axis = 1, inplace = True) # axis =1 determines column,
                                                    #inplace = True for change permanently in original data set

In [12]: data # printing to see the change in DataFrame
```

Out[12]:

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	2023	SE	FT	Data Scientist	85847	ES	100	ES	L
1	2023	MI	CT	Machine Learning	30000	US	100	US	S
2	2023	MI	CT	Machine Learning	25500	US	100	US	S
3	2023	SE	FT	Data Scientist	175000	CA	100	CA	M
4	2023	SE	FT	Data Scientist	120000	CA	100	CA	M
...	...	...	...	...	...	...	...	...	...
3750	2020	SE	FT	Data Scientist	412000	US	100	US	L
3751	2021	MI	FT	Data Scientist	151000	US	100	US	L
3752	2020	EN	FT	Data Scientist	105000	US	100	US	S
3753	2020	EN	CT	Data Analyst	100000	US	100	US	L
3754	2021	SE	FT	Data Science	94665	IN	50	IN	L

3755 rows × 9 columns

Figure 7 Dropping Columns

c. Write a python program to remove the NaN missing values from updated dataframe.

We use isnull() function to check if there is any null or NAN value. The output will be displayed in Boolean value which is shown below. It is either True of False with no parameter. If the Nan value is present, it will display True and if not, it will display False. But the data set consist of a lot of rows and column it is not clear with the output shown whether there is the presence of NaN value or not. So, we use another method which is mentioned below.

Write a python program to remove the NaN missing values from updated dataframe

```
In [13]: data.isnull() #Checking the null value is present or not using isnull function
```

Out[13]:

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...
3750	False	False	False	False	False	False	False	False	False
3751	False	False	False	False	False	False	False	False	False
3752	False	False	False	False	False	False	False	False	False
3753	False	False	False	False	False	False	False	False	False
3754	False	False	False	False	False	False	False	False	False

3755 rows × 9 columns

Figure 8 Checking null value

For knowing which columns have the exact number of NaN value we add sum() function after the isnull() function. The sum() function is used to returns the addition of all the value according to the user request. Therefore, the output shown in the figure displays the sum of NaN value in each column. All the columns have 0 NaN value present which means there is no NaN value in entire data set.

3755 rows × 9 columns

```
In [14]: data.isnull().sum() #Checking null value count in each column by adding sum function

Out[14]: work_year            0
         experience_level     0
         employment_type      0
         job_title            0
         salary_in_usd        0
         employee_residence   0
         remote_ratio         0
         company_location     0
         company_size         0
         dtype: int64
```

Figure 9 Checking null value count in each column

To remove NaN value from the DataFrame we again use drop function but with some additional features like dropna function which will only drop NaN. Dropna with how have two value any and all, if the value is all, only those rows or column with all value NaN will be deleted, but if there is a row or column where only one or more NaN value is present it will not delete, whereas when the value is any if there is presence of one, any or all value NaN it will delete the entire row or column. The default value of dropna is how = any we can either mention it or not, it does not matter. The NaN value is not present in entire DataFrame therefore the rows and column numbers will not be affected it remains the same after removing NaN value.

```
In [15]: data.dropna() #dropping NaN value using dropna function from the DataFrame
         data
```

Out[15]:

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	2023	SE	FT	Data Scientist	85847	ES	100	ES	L
1	2023	MI	CT	Machine Learning	30000	US	100	US	S
2	2023	MI	CT	Machine Learning	25500	US	100	US	S
3	2023	SE	FT	Data Scientist	175000	CA	100	CA	M
4	2023	SE	FT	Data Scientist	120000	CA	100	CA	M
...	...	...	...	...	...	...	...	...	...
3750	2020	SE	FT	Data Scientist	412000	US	100	US	L
3751	2021	MI	FT	Data Scientist	151000	US	100	US	L
3752	2020	EN	FT	Data Scientist	105000	US	100	US	S
3753	2020	EN	CT	Data Analyst	100000	US	100	US	L
3754	2021	SE	FT	Data Science	94665	IN	50	IN	L

3755 rows × 9 columns

Figure 10 Dropping null value

d. Write a python program to check duplicates value in the dataframe.

To check the duplicate value in the DataFrame pandas have a function called duplicated() which identify and displays the duplicate rows along with Boolean value which is True or False which is shown in the picture below. There are altogether 3755 rows so it is not possible to display all in the screen. We can check the total number of duplicated value in the DataFrame. The data is the variable where DataFrame is stored and dot means duplicated value of data variable.

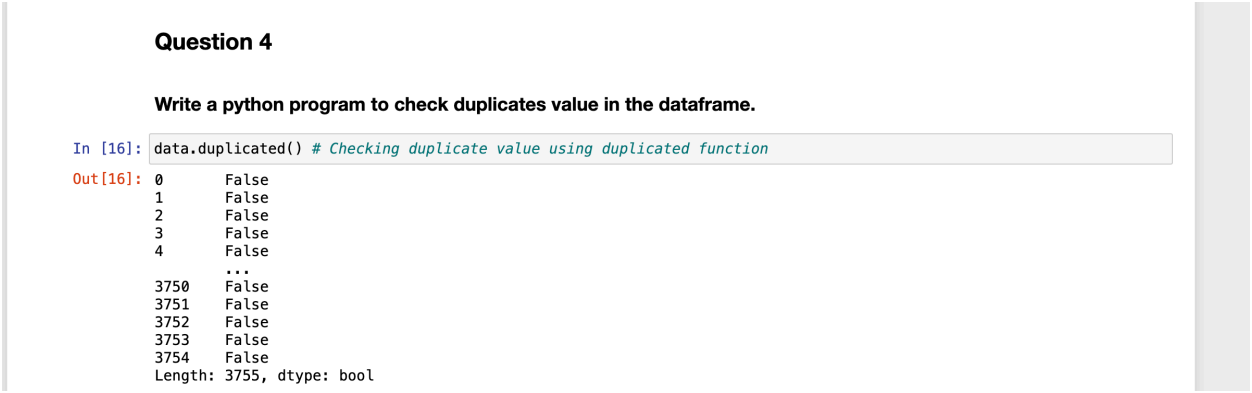


Figure 11 Checking duplicate value

The way to check the total duplicated value in the DataFrame is again using the sum function after the duplicated. It displays the total number of duplicated data which here is 1192.

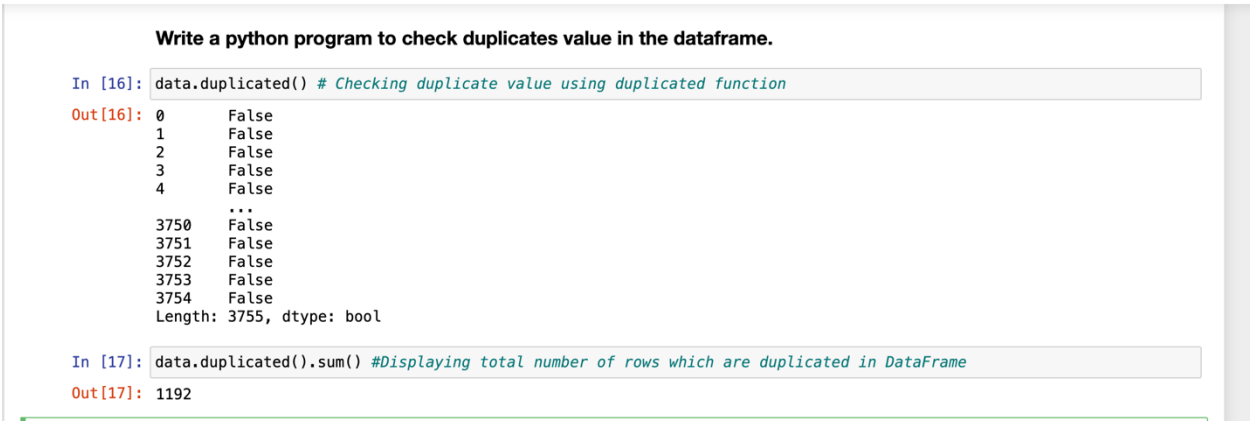


Figure 12 Checking duplicated value total number

But if we want to display the entire row of the duplicated data we need to use loc method because loc is specially use to retrieve group of rows and columns from the specified labels and condition. In this case loc is used to retrieve rows and column from the DataFrame(data) which are duplicate and ":" sign is given which means all the. mentioning or selecting all the column, we can also give column from our choice we just have to mention the name. Therefore, all the columns along with the rows are displayed which are duplicate in the DataFrame.

```
In [18]: data.loc[data.duplicated(), :] # Retreving only duplicated rows from all the column
```

Out[18]:

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
115	2023	SE	FT	Data Scientist	150000	US	0	US	M
123	2023	SE	FT	Analytics Engineer	289800	US	0	US	M
153	2023	MI	FT	Data Engineer	100000	US	100	US	M
154	2023	MI	FT	Data Engineer	70000	US	100	US	M
160	2023	SE	FT	Data Engineer	115000	US	0	US	M
...	...	...	...	...	...	...	...	...	...
3439	2022	MI	FT	Data Scientist	78000	US	100	US	M
3440	2022	SE	FT	Data Engineer	135000	US	100	US	M
3441	2022	SE	FT	Data Engineer	115000	US	100	US	M
3586	2021	MI	FT	Data Engineer	200000	US	100	US	L
3709	2021	MI	FT	Data Scientist	90734	DE	50	DE	L

1192 rows × 9 columns

Figure 13 Using loc method to retrive rows and column

Keeping the duplicated value is not a good practice. So to remove the duplicate data we use drop_duplicates() function. Drop duplicate function removes all the duplicated row. Here drop_duplicates() is used with parameter keep = first which keeps keeping the first row of each duplicated row and removing all the rows below, we can also give the value of keep last which will keep the last row of each duplicated row and remove all the row before. The default value is keep = first. The default value is inplace = false which means it will only take effect temporarily only that specific code is executed but to make the effect directly to the original DataFrame permanently we must mention true in the parameter. The output now shows 2563 rows which means from 3755 1192 has being deleted successfully.

```
In [19]: data.drop_duplicates(keep="first",inplace=True) #Deleting all the duplicated value and keeping the first occurence
         data
```

Out[19]:

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	2023	SE	FT	Data Scientist	85847	ES	100	ES	L
1	2023	MI	CT	Machine Learning	30000	US	100	US	S
2	2023	MI	CT	Machine Learning	25500	US	100	US	S
3	2023	SE	FT	Data Scientist	175000	CA	100	CA	M
4	2023	SE	FT	Data Scientist	120000	CA	100	CA	M
...	...	...	...	...	...	...	...	...	...
3750	2020	SE	FT	Data Scientist	412000	US	100	US	L
3751	2021	MI	FT	Data Scientist	151000	US	100	US	L
3752	2020	EN	FT	Data Scientist	105000	US	100	US	S
3753	2020	EN	CT	Data Analyst	100000	US	100	US	L
3754	2021	SE	FT	Data Science	94665	IN	50	IN	L

2563 rows × 9 columns

Figure 14 Dropping duplicate value

e. Write a python program to see the unique values from all the columns in the dataframe.

To print unique DataFrame have a function which is called unique(). But we cannot directly use this function in the Data where the dataframe is stored. The unique() function is used with specific columns to retrieve the unique values from that selected column. But to retrieve all the unique value from all the columns we can use for each loop to iterate over each column and apply unique function. First the code is executed in first column then its print the second print statement then it retrieves the unique value from that particular column again it executes the same process with another column, this continues again it reaches the last column and only stops if there are no columns left. Therefore, the output describe the unique value in each column of the dataframe.

Write a python program to see the unique values from all the columns in the dataframe.

```
In [21]: for column in data: #using for each loop in the column of dataframe to iterate over each column
             print("Unique Value in", column + ':') #printing to know which column value is being printed
             print (data[column].unique()) #retreiving unique values from column
```

```
Unique Value in work_year:
[2023 2022 2020 2021]
Unique Value in experience_level:
['SE' 'MI' 'EN' 'EX']
Unique Value in employment_type:
['FT' 'CT' 'FL' 'PT']
Unique Value in job_title:
['Data Scientist' 'Machine Learning ' 'Applied Scientist' 'Data Analyst'
 'Data Modeler' 'Research Engineer' 'Analytics Engineer'
 'Business Intelligence Engineer' 'Data Strategist' 'Data Engineer'
 'Computer Vision Engineer' 'Data Architect' 'AI Developer'
 'Research Scientist' 'Data Analytics' 'ETL Engineer'
 'Data DevOps Engineer' 'Head of Data' 'Data Science' 'Data Manager'
 'Data Specialist' 'MLOps Engineer' 'AI Scientist'
 'Autonomous Vehicle Technician' 'Cloud Database Engineer'
 'Data Infrastructure Engineer' 'AI Programmer' 'Data Operations Engineer'
 'BI Developer' 'Deep Learning Researcher' 'BI Analyst' 'Insight Analyst'
 'Deep Learning Engineer' 'Computer Vision Software Engineer' 'Data Lead'
 'NLP Engineer' 'Manager Data Management' '3D Computer Vision Researcher'
 'Data Management Specialist' 'Data Operations Analyst'
 'Power BI Developer' 'ETL Developer']
Unique Value in salary_in_usd:
[ 85847  30000  25500 ...  28369 412000  94665]
Unique Value in employee_residence:
['ES' 'US' 'CA' 'DE' 'GB' 'NG' 'IN' 'HK' 'PT' 'NL' 'CH' 'CF' 'FR' 'AU'
 'FI' 'UA' 'IE' 'IL' 'GH' 'AT' 'CO' 'SG' 'SE' 'SI' 'MX' 'UZ' 'BR' 'TH'
 'HR' 'PL' 'KW' 'VN' 'CY' 'AR' 'AM' 'BA' 'KE' 'GR' 'MK' 'LV' 'RO' 'PK'
 'IT' 'MA' 'LT' 'BE' 'AS' 'IR' 'HU' 'SK' 'CN' 'CZ' 'CR' 'TR' 'CL' 'PR'
 'DK' 'BO' 'PH' 'DO' 'EG' 'ID' 'AE' 'MY' 'JP' 'EE' 'HN' 'TN' 'RU' 'DZ'
 'IQ' 'BG' 'JE' 'RS' 'NZ' 'MD' 'LU' 'MT']
Unique Value in remote_ratio:
[100   0  50]
Unique Value in company_location:
['ES' 'US' 'CA' 'DE' 'GB' 'NG' 'IN' 'HK' 'NL' 'CH' 'CF' 'FR' 'FI' 'UA'
 'IE' 'IL' 'GH' 'CO' 'SG' 'AU' 'SE' 'SI' 'MX' 'BR' 'PT' 'RU' 'TH' 'HR'
 'VN' 'EE' 'AM' 'BA' 'KE' 'GR' 'MK' 'LV' 'RO' 'PK' 'IT' 'MA' 'PL' 'AL'
 'AR' 'LT' 'AS' 'CR' 'IR' 'BS' 'HU' 'AT' 'SK' 'CZ' 'TR' 'PR' 'DK' 'BO'
 'PH' 'BE' 'ID' 'EG' 'AE' 'LU' 'MY' 'HN' 'JP' 'DZ' 'IQ' 'CN' 'NZ' 'CL'
 'MD' 'MT']
Unique Value in company_size:
['L' 'S' 'M']
```

Figure 15 Using for each loop to retieve unique value from all column

f. Rename the experience level columns as below.

SE – Senior Level/Expert

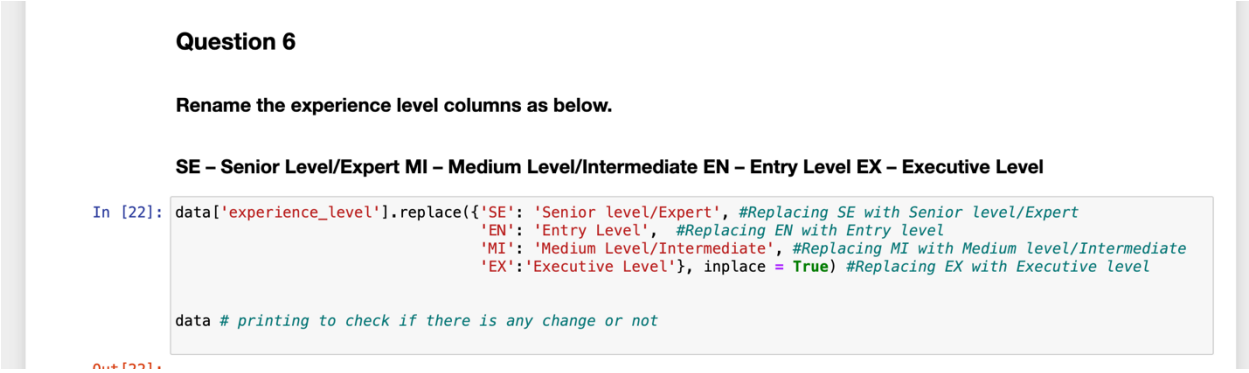
MI – Medium Level/Intermediate

EN – Entry Level

EX – Executive Level

To rename the value of column, we can use replace function of DataFrame. Here the column experience_level column is selected from the DataFrame to rename its values. The syntax follows a dictionary-like structure, where the original values are specified as keys and the replacement values are provided as corresponding values. The default value is inplace = false which means it will only take effect temporarily only that specific code is executed but to make the effect directly to the original DataFrame permanently we must mention true in the parameter.

The output displayed the replaced value or renamed value in experience level column we can compare it with before replacing the value which is shown below.



Question 6

Rename the experience level columns as below.

SE – Senior Level/Expert MI – Medium Level/Intermediate EN – Entry Level EX – Executive Level

In [22]:
```
data['experience_level'].replace({'SE': 'Senior level/Expert', #Replacing SE with Senior level/Expert
                                  'EN': 'Entry Level',  #Replacing EN with Entry level
                                  'MI': 'Medium Level/Intermediate', #Replacing MI with Medium level/Intermediate
                                  'EX':'Executive Level'}, inplace = True) #Replacing EX with Executive level


data # printing to check if there is any change or not
```

Figure 16 Replacing value in column

Out[22]:

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	2023	Senior level/Expert	FT	Data Scientist	85847	ES	100	ES	L
1	2023	Medium Level/Intermediate	CT	Machine Learning	30000	US	100	US	S
2	2023	Medium Level/Intermediate	CT	Machine Learning	25500	US	100	US	S
3	2023	Senior level/Expert	FT	Data Scientist	175000	CA	100	CA	M
4	2023	Senior level/Expert	FT	Data Scientist	120000	CA	100	CA	M
...	...	...	...	...	...	...	...	...	...
3750	2020	Senior level/Expert	FT	Data Scientist	412000	US	100	US	L
3751	2021	Medium Level/Intermediate	FT	Data Scientist	151000	US	100	US	L
3752	2020	Entry Level	FT	Data Scientist	105000	US	100	US	S
3753	2020	Entry Level	CT	Data Analyst	100000	US	100	US	L
3754	2021	Senior level/Expert	FT	Data Science	94665	IN	50	IN	L

2563 rows × 9 columns

Figure 17 Showing output after being replaced

4. Data Analysis

a. Write a Python program to show summary statistics of sum, mean, standard deviation, skewness, and kurtosis of any chosen variable.

Describe() function displays a summary statistic like count, mean, std, min, max along with 25%, 50% and 75%. Here only 3 columns are displayed because aggregated function only use columns with numeric values and by default describe function uses only numeric value strings cannot be used. Also, if there is mixed datatype like numeric and non numeric describe function only takes numeric value by default.

Question 1

Write a Python program to show summary statistics of sum, mean, standard deviation, skewness, and kurtosis of any chosen variable.

```
In [23]: data.describe() #showing summary stats of DataFrame
```

Out[23]:

	work_year	salary_in_usd	remote_ratio
count	2563.000000	2563.000000	2563.000000
mean	2022.298869	133167.557940	50.624268
std	0.750505	67206.729308	48.146768
min	2020.000000	5132.000000	0.000000
25%	2022.000000	84053.000000	0.000000
50%	2022.000000	129300.000000	50.000000
75%	2023.000000	175000.000000	100.000000
max	2023.000000	450000.000000	100.000000

Figure 18 Summary stats

But the question is about choosing a variable also showing showing skewness and kurtosis of the chosen variable. I selected the work_year column from the DataFrame and stored it in the variable name s. Now, instead of calling individual aggregate function separately like s.sum(), s.mean() etc I used dictionary-like structure, where the keys are the name of the aggregate function and value are the column from the database where we applied aggregate function. At last we displayed by printing the variable. The output is the aggregate function metioned in the question of the column work_year which is numeric because only numeric values can be calculated.

```
max  2023.000000  450000.000000  100.000000

In [25]: year = data['work_year']
         stats = {'sum':year.sum(), #showing  the sum of column work_year
                  'mean':year.mean(), #showing  the mean of column work_year
                  'standard deviation':year.std(), #showing  the standard deviation of column work_year
                  'skewness':year.skew(), #skewness of column work_year
                  'kurtosis':year.kurtosis() #kurtosis of column work_year
                 }

         stats #printing the variable
```

Figure 19 Summary stats for work year column

Sum is the total of all the values in the column, mean is the average of total of column, Standard deviation is calculated as the square root of variance. (HARGRAVE, 2023) Kurtosis is a measure of tailedness of a distribution and tailedness means how often outliers occur. (Turney, 2024) A skewness is a distribution of asymmetry of distribution. A distribution can have positive or negative or zero skwness. (Turney., 2023)

```
                 }

         stats #printing the variable

Out[25]: {'sum': 5183152,
          'mean': 2022.2988685134608,
          'standard deviation': 0.7505050910953022,
          'skewness': -0.966825202109312 3,
          'kurtosis': 0.7317174216970521}
```

Figure 20 Result of stats

b. Write a Python program to calculate and show correlation of all variables.

To calculate and show correlation of all variables Dataframe has a function called corr(). Correlation means if one variable data or value is increased it effect the other variable and increases the value of other variable as well at same rate same goes for decreased. If we apply corr function directly to dataframe it will display error because it cannot convert string to numeric or float. Therefore, to calculate the correlation we need to mention numeric_only = True which means it will now choose all the column with numeric value only and calculate and show the correlation variable. Here the correlation of column work_year, salary_in_usd and remote_ration is displayed because these 3 are the only numeric column in DataFrame.

Question 2

Write a Python program to calculate and show correlation of all variables.

```
In [26]: data.corr(numeric_only=True) # Displaying correlation of numeric value only because correlation cannot be shown for
```

Out[26]:

	work_year	salary_in_usd	remote_ratio
work_year	1.000000	0.236293	-0.215262
salary_in_usd	0.236293	1.000000	-0.082893
remote_ratio	-0.215262	-0.082893	1.000000

Figure 21 Showing Correlation

5. Data Exploration

a. Write a python program to find out top 15 jobs. Make a bar graph as well.

To identify the top 15 jobs we need to first count the value of each jobs how much time they have repeated. To do this there is a function called value_counts(). Value_counts function counts the occurrence of each unique value from the given Dataframe column. Head (15) function with parameter 15 is given because we are asked to find out the top 15 job. The default value of head function is 5 if we give no parameter in the head function it will display the first 5 rows. Therefore, the output below shows the occurrence of 15 unique job titles from the DataFrame. It displays the occurrence of unique job in descending order from highest to lowest.

Write a python program to find out top 15 jobs. Make a bar graph as well.

```
In [27]: top_15_job = data['job_title'].value_counts().head(15) #Counting the occurence of unique job title in the column and
         top_15_job # printing the output
```

Out[27]:

```
job_title
Data Engineer               625
Data Scientist              569
Data Analyst                450
Machine Learning            306
Data Science                108
Analytics Engineer           91
Data Architect               68
Research Scientist           65
Research Engineer            33
Applied Scientist            31
Data Analytics               30
Data Manager                 23
Computer Vision Engineer     18
AI Scientist                 16
Data Specialist              12
Name: count, dtype: int64
```

Figure 22 Displaying top 15 jobs

For Plotting Bar Graph of top 15 jobs, we use the plot function where the value of kind is bar which means it will display bar graph, the title is set to Top 15 jobs. Now we use alias we created while importing matplotlib and use xlabel and ylabel which is X axis and Y axis to identify what value they are representing. Legend is typically used only when there are multiple data set but I used it for better understanding.

```
In [47]: top_15_job.plot(kind='bar', title= 'Top 15 Jobs') # Plotting Bar Graph for top 15 jobs
         plt.xlabel('Jobs Title') # Giving the X-axis name Job title
         plt.ylabel('Value Counts') # Giving the Y-axis name Value Counts
         plt.legend()

Out[47]: <matplotlib.legend.Legend at 0x13c122d10>
```

Figure 23 Bar Graph of top 15 jobs

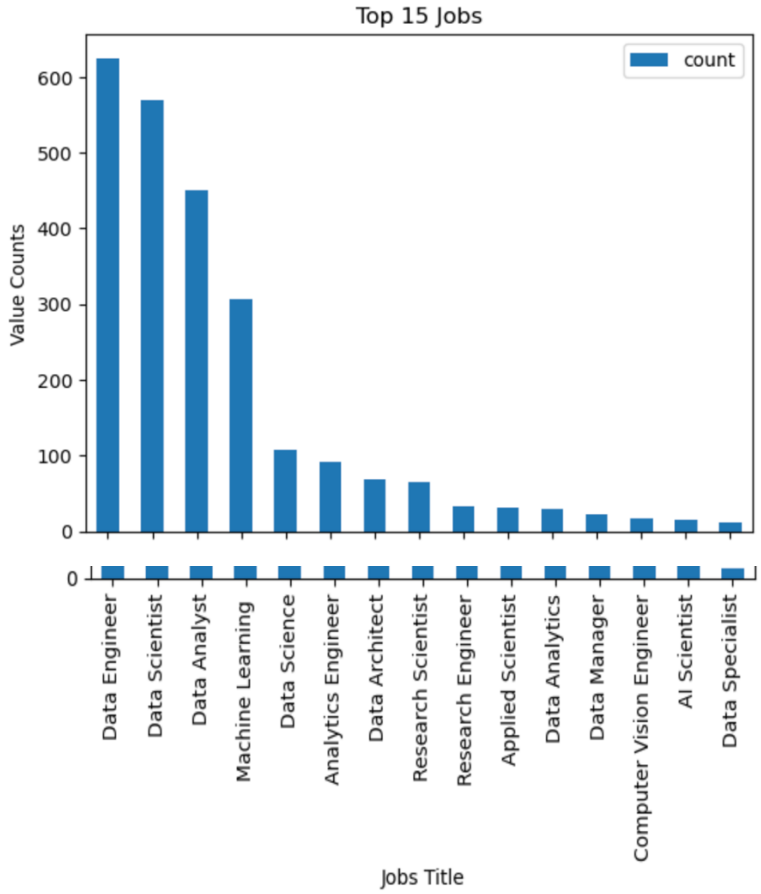


Figure 24 Bar Graph of top 15 jobs

b. **Which job has the highest salaries? Illustrate with bar graph**.

To display the highest salaries of according to the job at first we can use loc method, loc method is to retrieve rows and column value based on the condition. Here the loc is used to retrieve values from job_title and salary_in_usd columns and the condition is the data is grouped by job_title column and from the unique job retrieving the maximum salary based on the index where idxmax() function is used. This function is used to retrieve the index of row with maximum value of the given column. The group by function splits the data into group based on the criteria here the criteria is column name job_title. The output of this code is stored into max_salary variable because it will display the rows randomly, but we need it to be displayed from highest to lowest.

Now, we use sort_values function and the parameter consist of two value one is how to sort the value which here is based on salary_in_usd column and ascending = False means not from low to high, display from high to low. And I have decided to show only 15 rows so head function with parameter 15 is used to display the top 15 rows. Therefore, the output is now displayed grouped by job titles and sorted according by salaries which are the maximum salary of unique jobs.

Question 2

Which job has the highest salaries? Illustrate with bar graph.

In [48]:
```
max_salary = data.loc[data.groupby('job_title')['salary_in_usd'].idxmax(), ['job_title', 'salary_in_usd']]
# Using loc to retrieve only two column data which is grouped by job_title column
# And displaying on max salary from the unique job_tile with the help of index

max_salary = max_salary.sort_values(by='salary_in_usd', ascending=False)# sorting the values by salary_in_usd column
                                                                        #ascending = false which will display the highest to lowest
highest_salary = max_salary.head(15) # Only Displaying top 15
highest_salary #printing the value
```

Out[48]:

Figure 25 Showing highest salaries of job title

Out[48]:

	job_title	salary_in_usd
3522	Research Scientist	450000
2011	Data Analyst	430967
528	AI Scientist	423834
3747	Machine Learning	423000
3675	Data Scientist	416000
3463	Data Analytics	405000
649	Data Architect	376080
2359	Data Science	375000
1421	Applied Scientist	350000
33	Computer Vision Engineer	342810
228	Head of Data	329500
3410	Data Engineer	324000
83	AI Developer	300000
254	Research Engineer	293000
117	Analytics Engineer	289800

Figure 26 Output of highest salaries of job

For Plotting Bar Graph of job with highest salaries, we use the plot function where the value of kind is bar which means it will display bar graph, the title is set to Job having highest salary and the color is red the value of X is given job_title because it will display the index if x value is not set. Now we use alias we created while importing matplotlib and use xlabel and ylabel which is X axis and Y axis to identify what value they are representing. Legend is typically used only when there are multiple data set but I used it for better understanding.

254	Research Engineer	293000
117	Analytics Engineer	289800

In [70]:
```
highest_salary.plot(kind='bar',x='job_title', title= 'Job Having Highest Salary', color='red')
# Displaying Bar Graph with giving the x value because it will display index in x axis along with title and differen
plt.xlabel('Jobs') # Giving the X-axis name Jobs
plt.ylabel('Salaries') # Giving the Y-axis name Salary
plt.legend() # Display legend
```

Out[70]: <matplotlib.legend.Legend at 0x15aaa2010>

Job Having Highest Salary

Figure 27 Code for bar graph of highest salaries job

```
Out[70]: <matplotlib.legend.Legend at 0x15aaa2010>
```

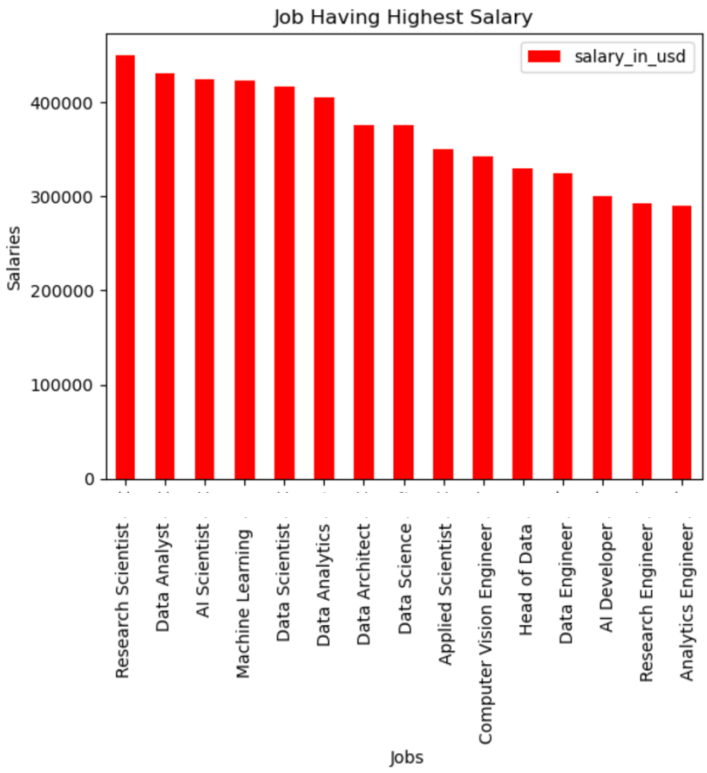
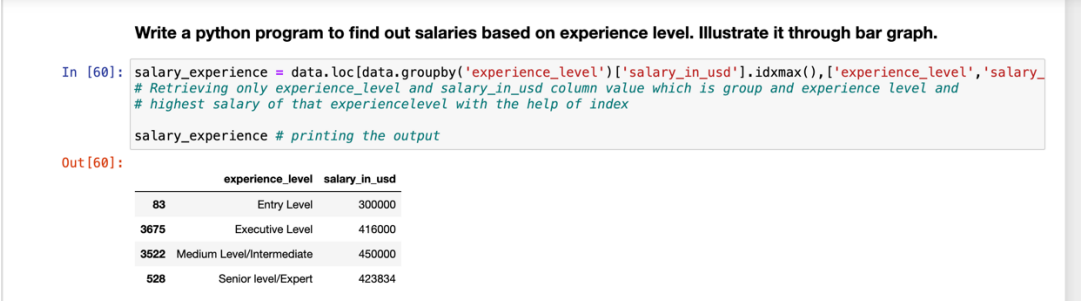


Figure 28 Bar graph of highest salaries job

c. Write a python program to find out salaries based on experience level. Illustrate it through bar graph.

To display the salaries based on experience level we can use loc method, loc method is to retrieve rows and column value based on the condition. Here the loc is used to retrieve values from experience_level and salary_in_usd columns and the condition is the data is grouped by experience_level column and from the unique experience level retrieving the maximum salary based on the index where idxmax() function is used. This function is used to retrieve the index of row with maximum value of the given column. The group by function splits the data into group based on the criteria here the criteria is column name job_title. The output of this code is stored into salary_experience. Therefore, the output is 4 rows because experience level holds 4 unique value and from it displaying the maximum salary.



Write a python program to find out salaries based on experience level. Illustrate it through bar graph.

In [60]:
```
salary_experience = data.loc[data.groupby('experience_level')['salary_in_usd'].idxmax(),['experience_level','salary_
# Retrieving only experience_level and salary_in_usd column value which is group and experience level and
# highest salary of that experiencelevel with the help of index

salary_experience # printing the output
```

Out[60]:

	experience_level	salary_in_usd
83	Entry Level	300000
3675	Executive Level	416000
3522	Medium Level/Intermediate	450000
528	Senior level/Expert	423834

Figure 29 Retrieving max salary based on experience level

For Plotting Bar Graph of highest salaries based on experience level , we use the plot function where the value of kind is bar which means it will display bar graph, the title is set to salary based on experience level and the color is green and the value of X is given experience level because it will display the index if x value is not set. Now we use alias we created while importing matplotlib and use xlabel and ylabel which is X axis and Y axis to identify what value they are representing. Here the x axis represents the index of Experience level and Y axis represent Salary Legend is typically used only when there are multiple data set but I used it for better understanding.

```
528          Senior level/Expert          423834

In [71]: salary_experience.plot(kind='bar', x = 'experience_level', title = "Salary Based on experience level", color="green"
         # Displaying Bar Graph with giving the x value because it will display index in x axis along with title and differen
         plt.xlabel('Experience_Level') # Giving the X-axis name Experience Level
         plt.ylabel('Salary') # Giving the Y-axis name Salary
         plt.legend()

Out[71]: <matplotlib.legend.Legend at 0x15ab0c0d0>
```

Figure 30 Code for Bar Graph of max salary based on experience level

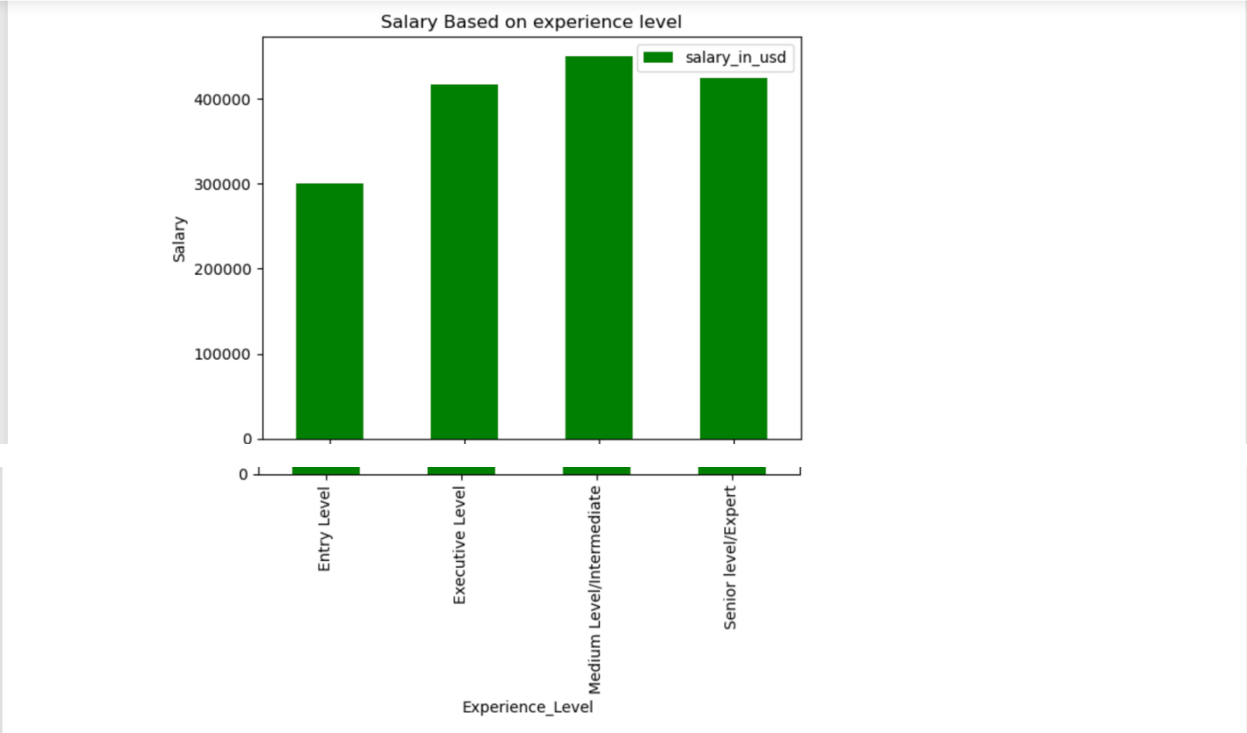


Figure 31 Bar Graph of max salary based on experience level

d. Write a Python program to show histogram and box plot of any chosen different variables. Use proper labels in the graph.

The fig is the figure where two figure exist one Is figure 1 and another is figure 2, subplots is the function of matplotlib where the parameters are nrows = 1 means number of rows is 1 and ncols=2 means number of columns are two and the figsize = (12,4) which means 12 inches width and 4 inches height. Bins refers to the intervals into which data is divided. Here the length of bins are equal to the length of unique values In work_year column which means there are 4 different unique value in work_year so it will display 4 different value along with their occurrence or value count. The edge color is set to black to differentiate the bar and color is set red. The title to year histogram and x axis to year and y axis to value count.

Same goes with figure 2 hist function is used to plot the histogram and it is based upon the salary_in_usd column with edge color black and color red to differentiate two different histograms. Setting the x axis to salary and y axis to value count because it displays the x axis value as salary and y axis value as value count.

```
In [64]: fig, (figure_1, figure_2) = plt.subplots(nrows=1, ncols=2, figsize=(12, 4))
         # Creating a figure with two subplots arranged horizontally

         figure_1.hist(data['work_year'], bins=len(data['work_year'].unique()), edgecolor='black', color='red')
         # Plotting a histogram of the 'work_year' column in the first subplot

         figure_1.set_title('Year Histogram')
         # Setting title for the first subplot

         figure_1.set_xlabel('Year') # Giving the X-axis name Year
         figure_1.set_ylabel('Value Counts') # Giving the Y-axis name Value Counts

         figure_2.hist(data['salary_in_usd'], edgecolor='black',color='green')
         # Plotting a histogram of the 'salary_in_usd' column in the second subplot

         figure_2.set_title('Salary Histogram')
         # Setting title for the second subplot

         figure_2.set_xlabel('Salary in USD') # Giving the X-axis name Salary in Usd
         figure_2.set_ylabel('Value Count') # Giving the Y-axis name Value Counts
Out[64]: Text(0, 0.5, 'Value Count')
```

Figure 32 Code for histogram of work year and salary in usd column

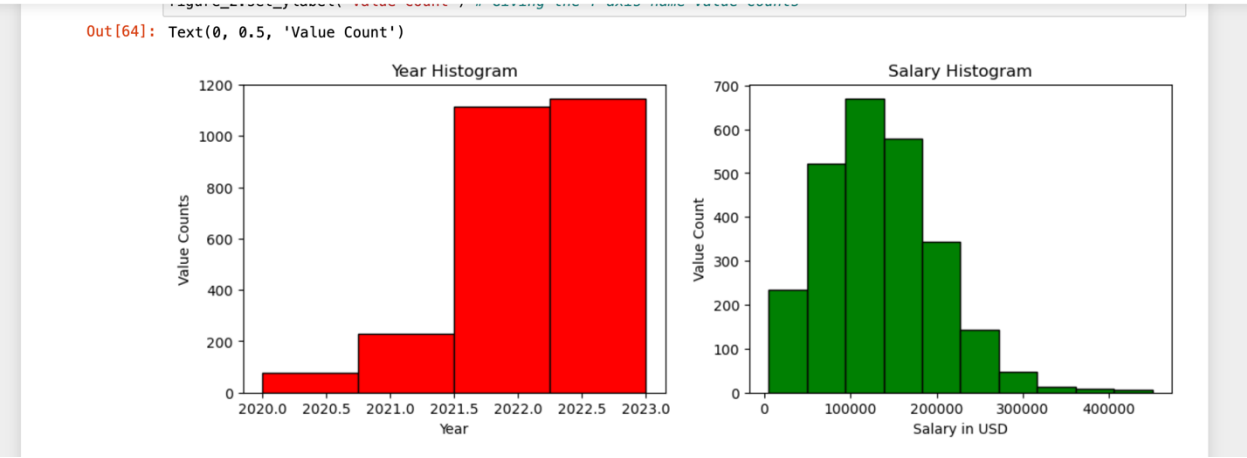


Figure 33 Histogram of work year and salary in usd column

To plot boxplot, boxplot function is used. Here the output is displayed the boxplot of the salary in usd column. The title, x-axis and y-axis are labelled for the clarification. Title is Boxplot of Salary in USD, xlabel is Salary Distribution, and ylabel is Salary(USD)

```
In [75]: plt.boxplot(data['salary_in_usd']) # Creating box plot
         plt.title('Boxplot of Salary in USD') # Giving the title name Salary in USD
         plt.xlabel('Salary Distribution') # Giving the X-axis name Salary distribution
         plt.ylabel('Salary (USD)') # Giving the Y-axis name Salary (USD)

Out[75]: Text(0, 0.5, 'Salary (USD)')
```

Figure 34 Code for Boxplot

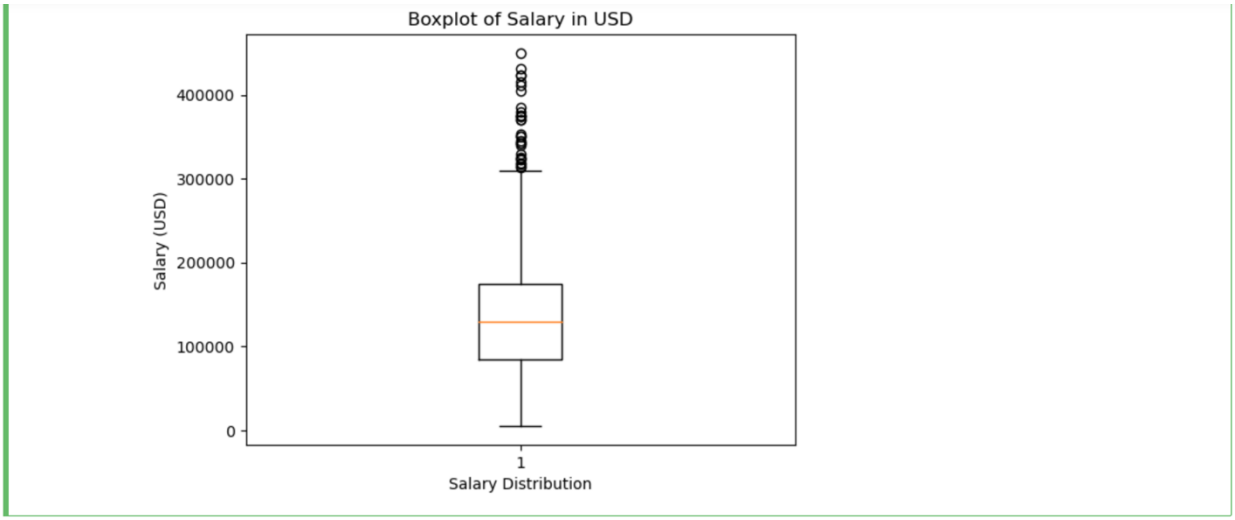


Figure 35 Box plot of salary in usd column

To plot boxplot, boxplot function is used. Here the output is displayed the boxplot of the work year column. The title, x-axis and y-axis are labelled for the clarification. Title is Boxplot of work year , xlabel is year , and ylabel is work year.

```
In [82]: plt.boxplot(data['work_year']) # Creating box plot
         plt.title('Boxplot of work year') # Giving the title name Salary in USD
         plt.xlabel('year') # Giving the X-axis name Salary distribution
         plt.ylabel('work year') # Giving the Y-axis name Salary (USD)

Out[82]: Text(0, 0.5, 'work year')
```

Figure 36 code for box plot of work year column

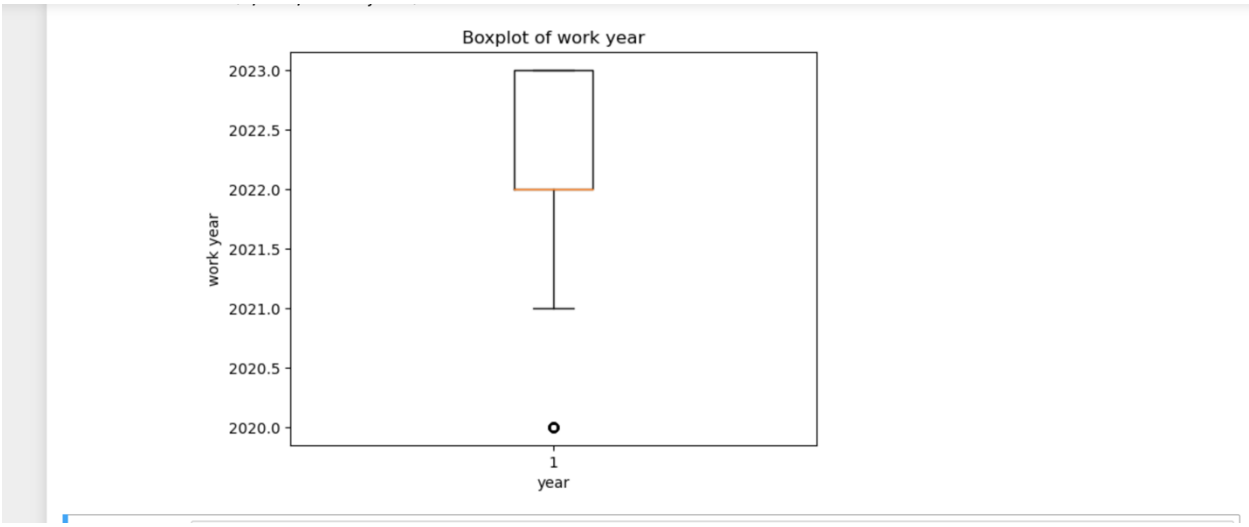


Figure 37 Box plot of work year column

Conclusion

From the coursework we have learn many new things about data manipulation and data execution. We have learned library such as pandas and matplotlib and their various functions. Function likes null, unique, duplicated, replace, drop etc and methods like loc to retrieve rows and column, using various aggregate function like sum, mean, sd, kurtosis to calculate summary stats of the numeric value column, displaying visual representation of given condition like bar graph, histogram and boxplot.

It also gave a glimpse of real word project how the things are carried out. Completing the coursework has somehow increased my research skill and documentation skills. It also has helped in managing time.

There were some challenging faced while completing the coursework, but it was all solved by consulting with our tutors and lecturer and reviewing the progress every week and the suggestion from them has helped to make the coursework best.

References

HARGRAVE, M., 2023. *Standard Deviation Formula and Uses vs. Variance.* [Online]
Available at: https://www.investopedia.com/terms/s/standarddeviation.asp
[Accessed 5 May 2024].

Turney, S., 2024. *What Is Kurtosis? | Definition, Examples & Formula.* [Online]
Available at:
https://www.scribbr.com/statistics/kurtosis/#:~:text=Kurtosis%20is%20a%20measure%20of,(medium%20tails)%20are%20mesokurtic.
[Accessed 3 May 2024].

Turney., S., 2023. *Skewness | Definition, Examples & Formula.* [Online]
Available at: https://www.scribbr.com/statistics/skewness/
[Accessed 3 May 2024].

Geeks, G. f., 2024. *Pandas Introduction.* [Online]
Available at: https://www.geeksforgeeks.org/introduction-to-pandas-in-python/
[Accessed 3 May 2024].

matoplotlib, 2023. *Matplotlib: Visualization with Python.* [Online]
Available at: https://matplotlib.org/
[Accessed 3 May 2024].