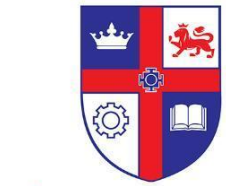
LONDON METROPOLITAN UNIVERSITY



islington college
(इस्लिङ्टन कलेज)

CC5067NI-Smart Data Discovery

60% Individual Coursework

2023-24 Spring

Student Name: Anjan Giri

London Met ID: 22085496

College ID: NP01CP4S230104

Assignment Due Date: Monday, May 13, 2024

Assignment Submission Date: Sunday, May 12, 2024

Word Count: 2526

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. Data Understanding 1

2. Data Preparation 4

2.1 Write a python program to load data into pandas DataFrame 4

2.2 Write a python program to remove unnecessary columns i.e., salary and salary currency. 5

2.3 Write a python program to remove the NaN missing values from updated dataframe. 8

2.4 Write a python program to check duplicates value in the dataframe. 9

2.5 Write a python program to see the unique values from all the columns in the dataframe. 10

2.6 Rename the experience level columns as below: 13

3. Data Analysis 14

3.1 Write a Python program to show summary statistics of sum, mean, standard deviation, skewness, and kurtosis of any chosen variable. 14

3.2 Write a Python program to calculate and show correlation of all variables. 17

4. Data Exploration 18

4.1 Write a python program to find out top 15 jobs. 18

4.2 Which job has the highest salaries? Illustrate with bar graph. 19

4.3 Write a python program to find out salaries based on experience level. Illustrate it through bar graph. 21

4.4 Write a Python program to show histogram and box plot of any chosen different variables. Use proper labels in the graph. 23

Histograms 23

Boxplots 25

5. Conclusion 27

Table Of Figures

Figure 1: importing pandas in the python file and reading the csv file in the pandas dataframe 4
Figure 2: printing the data stored in the 'salaries' dataframe 4
Figure 3: columns present in the data 5
Figure 4: dropping the unnecessary columns 5
Figure 5: displaying columns present in the data 6
Figure 6: entering code to drop the columns in the original DataFrame 6
Figure 7: original DataFrame is updated 7
Figure 8: sum of the number of null values present in the DataFrame 8
Figure 9: removing NaN values present in the DataFrame 8
Figure 10: checking for duplicate values 9
Figure 11: finding out the unique values using for loop 10
Figure 12: unique values (part-1) 11
Figure 13: unique values (part-2) 12
Figure 14: defining the mapping of old values to new values 13
Figure 15: renaming the required data 13
Figure 16: checking the values in 'salary_in_usd' column 14
Figure 17: sum of all values present in the column 14
Figure 18: calculating mean value of the column 15
Figure 19: calculating standard deviation of the column 15
Figure 20: calculating skewness of the column 16
Figure 21: calculating kurtosis value of the column 16
Figure 22: excluding columns with String values 17
Figure 23: finding correlation of the columns 17
Figure 24: finding top 15 jobs 18
Figure 25: calculating highest salaries 19
Figure 26: plotting the calculated values onto a bar graph 19
Figure 27: bar graph of jobs with highest salaries 20

Figure 28: calculating salaries based on experience level 21
Figure 29: plotting the calculated salaries onto a bar graph 21
Figure 30: bar graph of salaries based on experience levels 22
Figure 31: histogram of salary_in_usd column 23
Figure 32: histogram of work_year column 24
Figure 33: plotting the column salary_in_usd into a boxplot 25
Figure 34: boxplot of salary_in_usd column 25
Figure 35: plotting the column work_year into a boxplot 26
Figure 36: boxplot of work_year column 26

Table of Tables

Table 1: columns present in the dataset 3

1. Data Understanding

Data understanding includes thoroughly examining a dataset to comprehend its components, quality, and significance. This involves jumping into the data to distinguish patterns, relationships, and abnormalities, while also evaluating its suitability for analysis. Strategies such as data profiling, visualization, and exploratory analysis are typically utilized to uncover insights that inform further analytical processes.

In the given coursework, we are provided with a dataset to work with for learning and sharpening our knowledge of data, datasets and how to work on those data. The dataset consists of various columns providing information about the salaries that are provided to different data science related jobs. The columns present in the dataset are:

work_year, experience_level, employement_type, job_title, salary, salary_currency, salary_in_usd, employee_residence, remote_ratio, company_location and company_size. Each column provides data assigned to them and gives a proper structure to the data of the salaries of the data science employees along with their locations and job titles with their experience levels. Short description of columns present in the dataset are provided below: -

S.N.	Column Name	DataType	Description
1.	work_year	int64	This column provides information of the year on which all other data present in the dataset are based upon.
2.	experience_level	object	The experience_level column consists of four values SE, MI, EN and EX which gives

			the information about the experience level of the employees.
3.	employment_type	object	This column also consists of four values FT, CT, FL and PT which gives the employment type of the employees.
4.	job_title	object	The job_title column gives the name of the jobs that the employees are hired into.
5.	salary	int64	Salary column provides the data on the salary provided to each employee.
6.	salary_currency	object	This column states the currency of the salary that the employees are provided.
7.	salary_in_usd	int64	The salary_in_usd column gives the salaries of all the employees in USD value.
8.	employee_residence	object	This column gives the data of the residences of the employees.
9.	remote_ratio	int64	remote_ratio column gives the ratio of the work performed remotely in the particular job title.

10.	company_location	object	This column provides the country name where the company is situated.
11.	company_size	object	The company_size column gives the data on the size of the company: whether large (L), medium (M) or small (S).

Table 1: columns present in the dataset

2. Data Preparation

2.1 Write a python program to load data into pandas DataFrame

```
[200]: import pandas as pd #importing pandas by giving the alias 'pd'
       import matplotlib.pyplot as plt #importing matplotlib
       %matplotlib inline
       import csv

[202]: salaries = pd.read_csv("DataScienceSalaries.csv") #reading the csv file in pandas dataframe
```

Figure 1: importing pandas in the python file and reading the csv file in the pandas dataframe

Firstly, to work with the data in .csv file and to work with it, we need to import pandas which is an open-source library that provides data structures and data analysis tools in python.

After that the csv file is read in 'salaries' DataFrame using the code: pd.read_csv("file_name")

```
[133]: salaries.head() #printing the data stored in the 'salaries' dataframe
```

[133]:

	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	Principal Data Scientist	85847	ES	100	ES	L
1	2023	Medium Level/Intermediate	CT	ML Engineer	30000	US	100	US	S
2	2023	Medium Level/Intermediate	CT	ML Engineer	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

Figure 2: printing the data stored in the 'salaries' dataframe

Here, the csv file is read by the program and its data are stored in the 'salaries' dataframe.

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

```
*[13]: salaries.columns #displaying the columns

[13]: Index(['work_year', 'experience_level', 'employment_type', 'job_title',
       'salary', 'salary_currency', 'salary_in_usd', 'employee_residence',
       'remote_ratio', 'company_location', 'company_size'],
      dtype='object')
```

Figure 3: columns present in the data

The columns present in the data of DataScienceSalaries.csv file are displayed using '.columns' function.

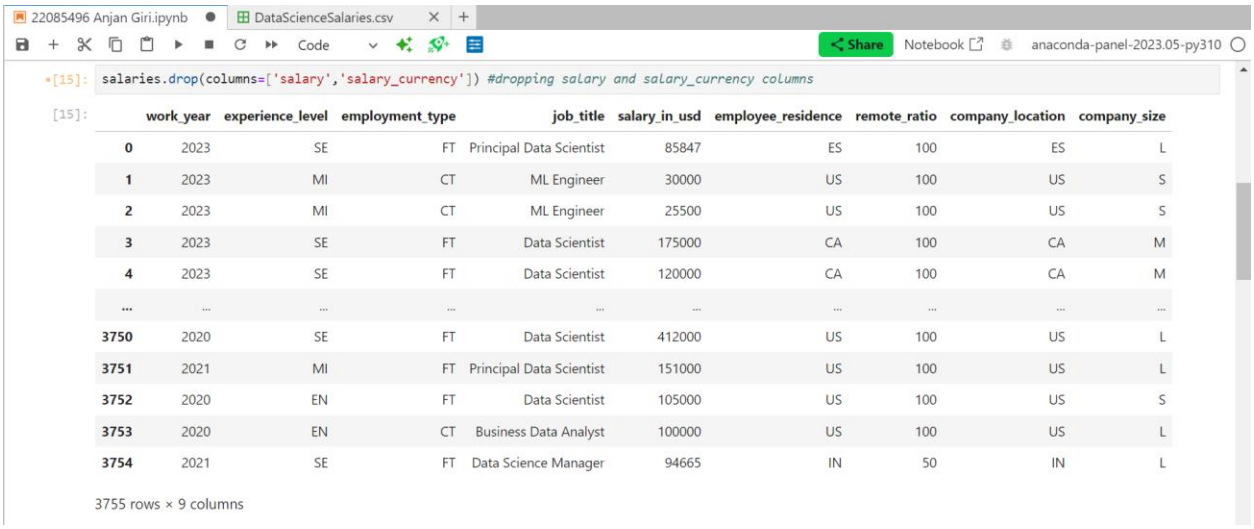


Figure 4: dropping the unnecessary columns

Both 'salary' and 'salary_currency' columns are removed by using '.drop' function.

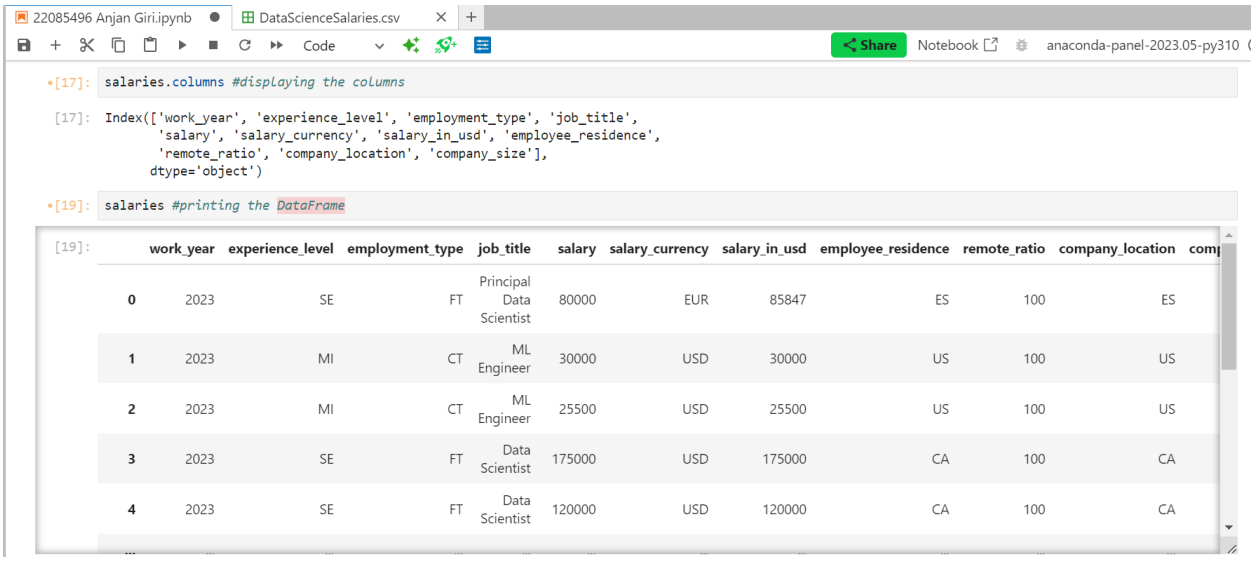


Figure 5: displaying columns present in the data

Although the columns were removed in Figure: 4 using '.drop' function, the original DataFrame has not changed as seen in Figure: 5.

To change the original DataFrame, 'inplace = True' is used in the code: "salaries.drop(columns=['salary','salary_currency'], inplace = True)"

Here, using the code "inplace=True" means that the operation will be executed in the DataFrame itself and the changes will be made in the original DataFrame without creating its copies.

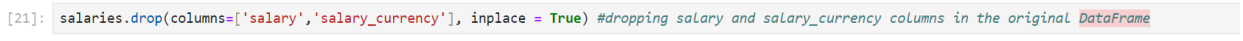
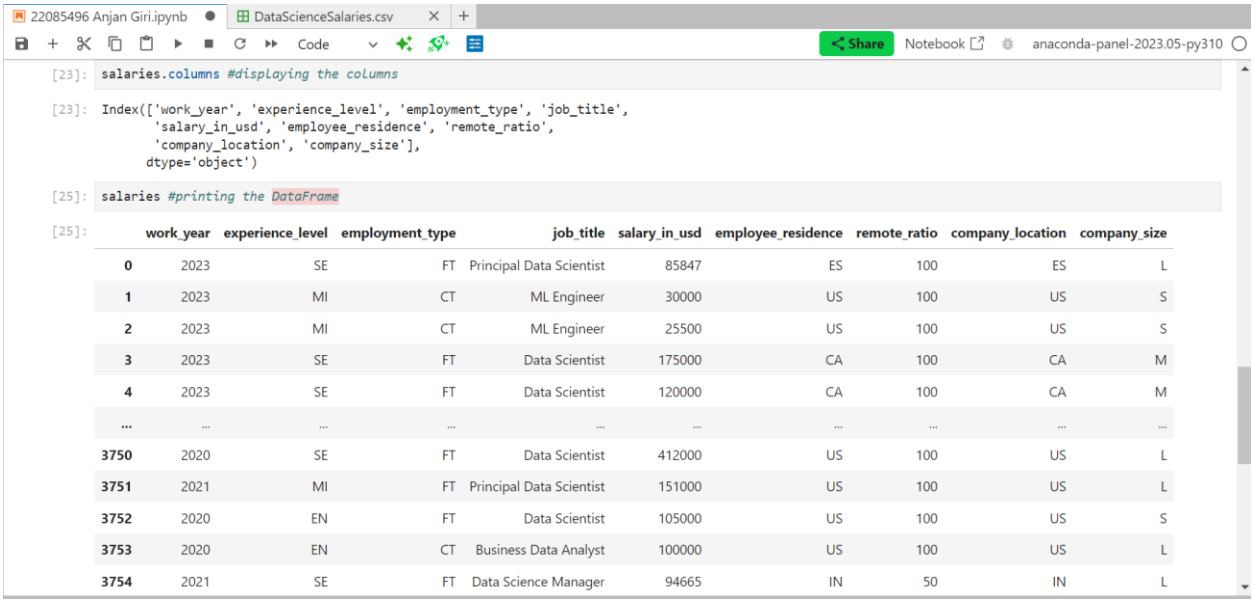


Figure 6: entering code to drop the columns in the original DataFrame



	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	2023	SE	FT	Principal Data Scientist	85847	ES	100	ES	L
1	2023	MI	CT	ML Engineer	30000	US	100	US	S
2	2023	MI	CT	ML Engineer	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	Principal Data Scientist	151000	US	100	US	L
3752	2020	EN	FT	Data Scientist	105000	US	100	US	S
3753	2020	EN	CT	Business Data Analyst	100000	US	100	US	L
3754	2021	SE	FT	Data Science Manager	94665	IN	50	IN	L

Figure 7: original DataFrame is updated

After executing the code in Figure: 6 the columns are removed in the original DataFrame.

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

```
[30]: salaries.isnull().sum() #number of missing values in the DataFrame

[30]: 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 8: sum of the number of null values present in the DataFrame

The above command is used to find out the sum of number of null values present in the DataFrame and we can see that there are no null values present in the DataFrame.

```
[36]: salaries.dropna(inplace=True) #removing NaN values (if any present in the DataFrame)
```

Figure 9: removing NaN values present in the DataFrame

Although no null values are present in the DataFrame, the code to remove the NaN values is executed in Figure: 9.

Here, '.dropna()' function is used to remove the NaN values present in the DataFrame.

'inplace=True' is used along with '.dropna' function so that the NaN values are removed in the original DataFrame.

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

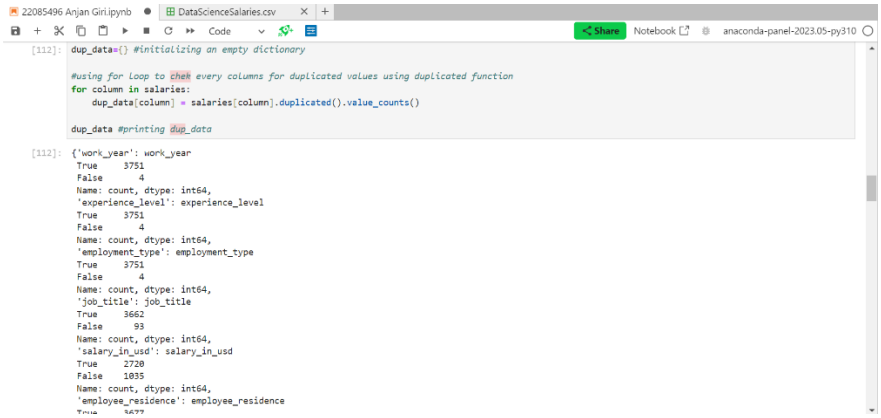


Figure 10: checking for duplicate values

First, a new variable ‘dup_data’ is initialized. Then DataFrame ‘salaries’ is checked for any duplicate values present by using ‘.duplicated’ function using for loop to check every columns for duplicate values. Lastly, the duplicated values stored in the variable are printed out.

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

```
[216]: notNumericColumns = salaries.select_dtypes(exclude = ['int64']).columns #excluding the columns containing numeric values

un_data = {} #initializing an empty dictionary

#using for loop to check unique values in every columns stored in notNumericColumns using .unique function
for column in notNumericColumns:
    un_data[column] = salaries[column].unique()

un_data #printing un_data
```

Figure 11: finding out the unique values using for loop

The columns containing numeric values are excluded and remaining columns are stored in the variable ‘noNumericColumns’. Then, a new dictionary is initialized. For loop is used to check the columns present in ‘notNumeriColumns’ variable and the unique values are found using ‘.unique’ function. The unique values are stored in the ‘un_data’ dictionary.

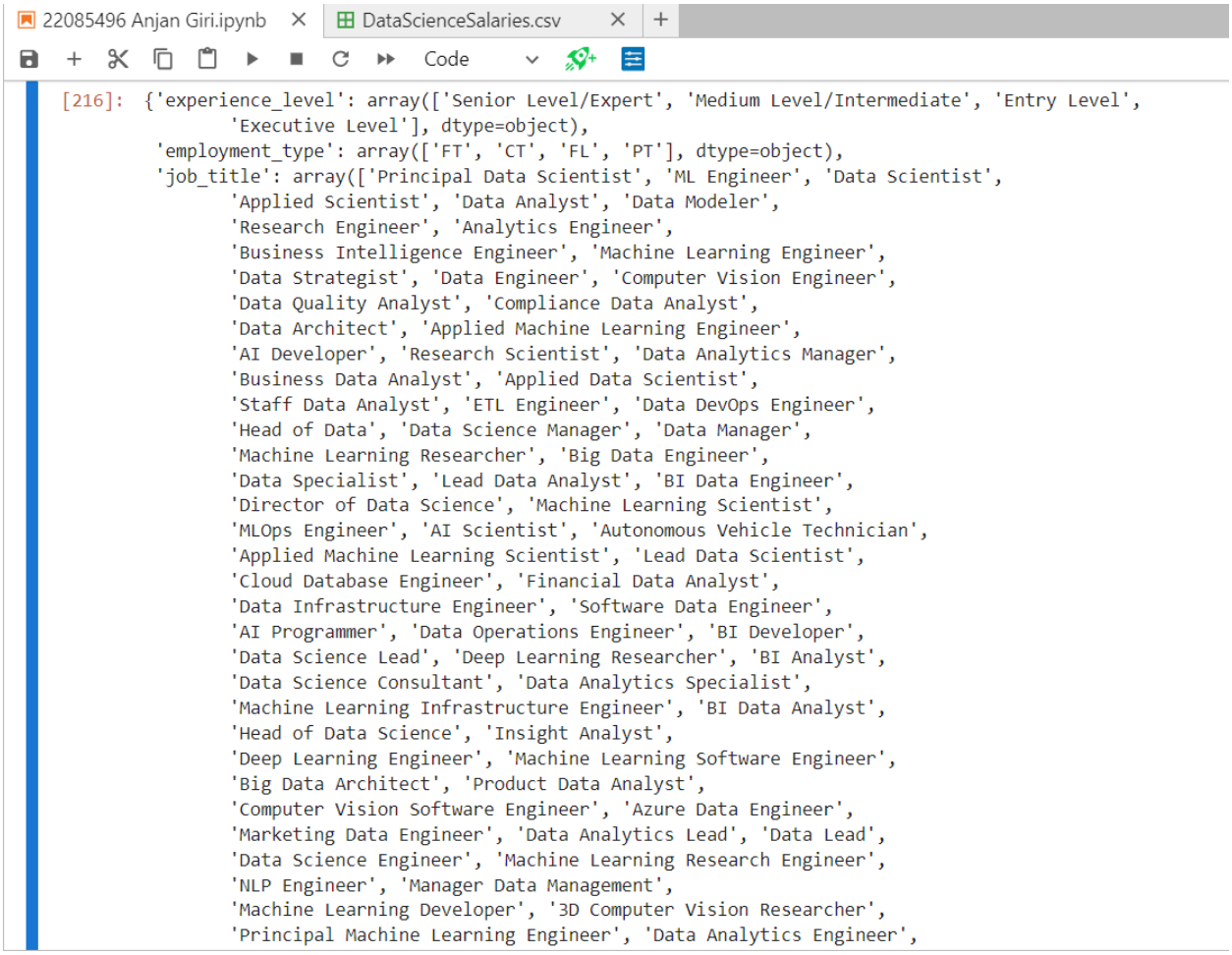


Figure 12: unique values (part-1)

The unique values stored in ‘un_data’ are displayed.

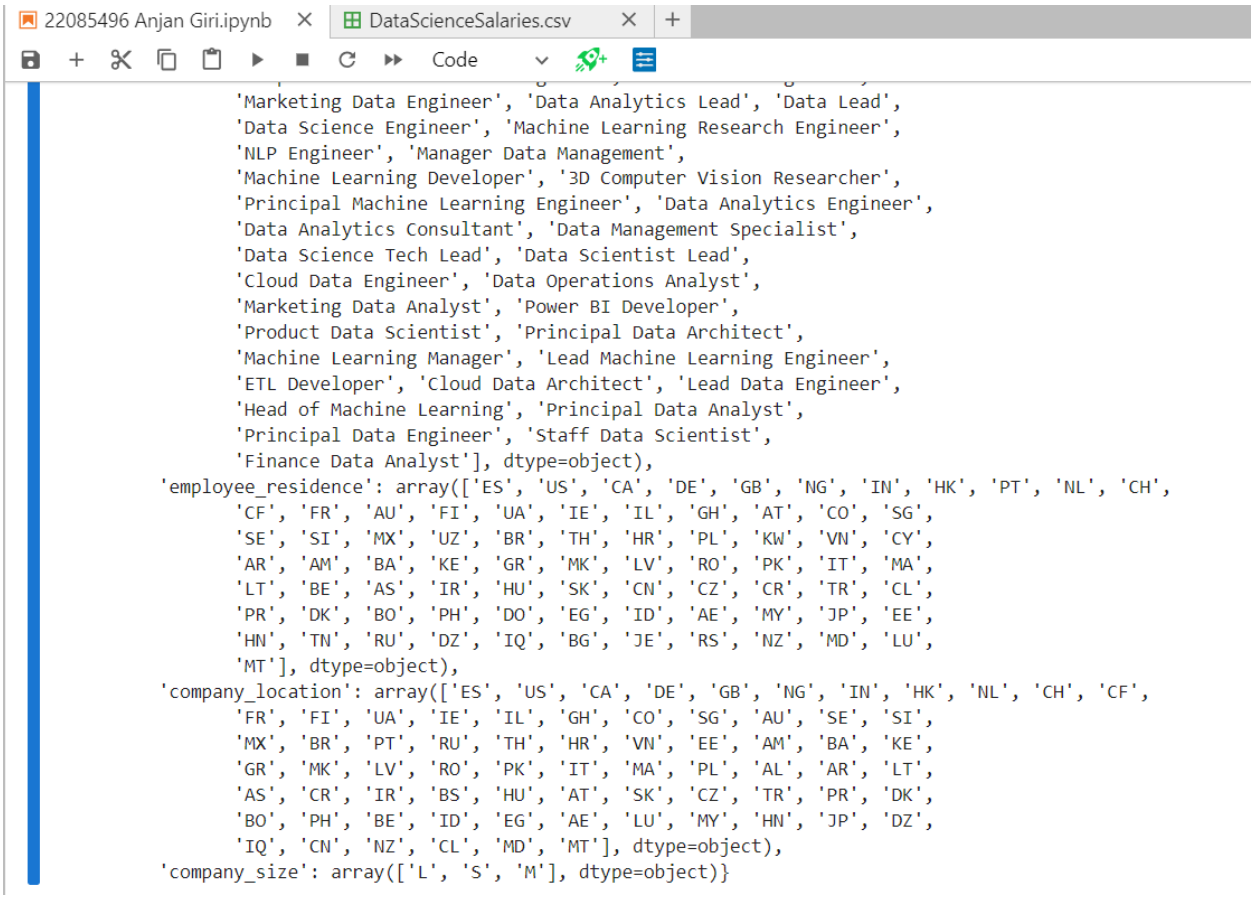


Figure 13: unique values (part-2)

The unique values stored in ‘un_data’ are displayed.

2.6 Rename the experience level columns as below:

SE – Senior Level/Expert

MI – Medium Level/Intermediate

EN – Entry Level

EX – Executive Level

```
[63]: #defining the conversion of old value to new value
rename_data = {
    'SE' : 'Senior Level/Expert',
    'MI' : 'Medium Level/Intermediate',
    'EN' : 'Entry Level',
    'EX' : 'Executive Level'
}
```

Figure 14: defining the mapping of old values to new values

A 'rename_data' dictionary is created by defining the conversion of old values to new values.

```
[65]: salaries['experience_level'] = salaries['experience_level'].replace(rename_data) #renaming the values in the 'experience_level' column
[67]: salaries
```

[67]:

	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	Principal Data Scientist	85847	ES	100	ES	L
1	2023	Medium Level/Intermediate	CT	ML Engineer	30000	US	100	US	S
2	2023	Medium Level/Intermediate	CT	ML Engineer	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

```
[ ]:
```

Figure 15: renaming the required data

The data present in the 'experience_level' column are renamed by using '.replace' function and the dictionary defined in Figure: 12.

3. Data Analysis

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

```
[81]: salaries.salary_in_usd #printing the data of 'salary_in_usd' column

[81]: 0        85847
      1        30000
      2        25500
      3       175000
      4       120000
               ...
      3750    412000
      3751    151000
      3752    105000
      3753    100000
      3754     94665
      Name: salary_in_usd, Length: 3755, dtype: int64
```

Figure 16: checking the values in 'salary_in_usd' column

```
[83]: sumOfSalary = sum(salaries.salary_in_usd) #adding the values present in whole column

[85]: sumOfSalary #printing the sum of the values

[85]: 516576814
```

Figure 17: sum of all values present in the column

Here, the values of the ‘salary_in_usd’ column are added and stored in the ‘sumOfSalary’ variable.

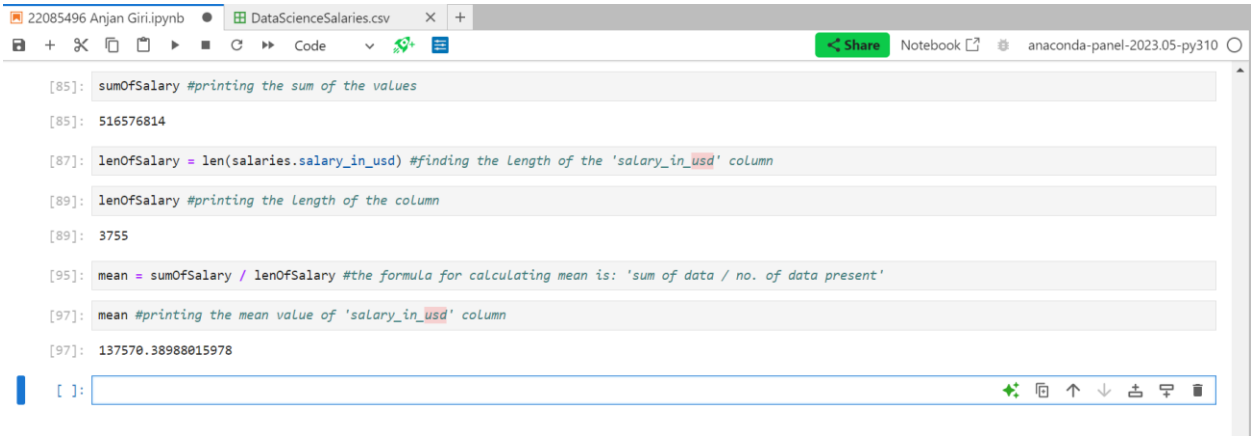


Figure 18: calculating mean value of the column

To calculate mean value of the column, the formula is used where sum of the values present in the column is divided by the number of values present in the column, i.e. length of the 'salary_in_usd' column.

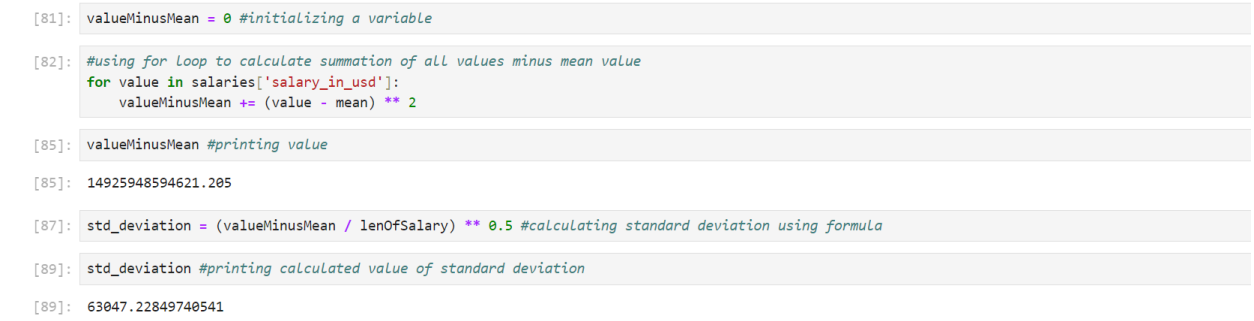


Figure 19: calculating standard deviation of the column

For calculating standard deviation, firstly the mean value of the column is subtracted from all values present in the column and the subtracted value is squared. Then, each of those calculated values are added and stored in the variable 'valueMinusMean' (Here, for loop is used to perform the calculation in every value of the column'). After that, the formula to calculate standard deviation is used where the calculated value in the variable 'valueMinusMean' is divided by the length of the column and the whole value is multiplied by the power of 0.5 (root).

```
[232]: valueForSkewness = 0 #initializing a variable

[234]: skewnessSummationValue = (salaries['salary_in_usd'] - mean) ** 3 #calculating value used in the formula for calculating skewness

       #using for loop to calculate the values for 'valueForSkewness' using every data present in the column
       for values in skewnessSummationValue:
           valueForSkewness += values

       valueForSkewness #printing the calculated value of 'valueForSkewness'

[234]: 5.045731243525176e+17

[236]: skewness = valueForSkewness / ((lenOfSalary - 1) * (std_deviation ** 3)) #using the formula for calculating skewness

[238]: skewness #printing the calculated value of skewness

[238]: 0.5363296982353364
```

Figure 20: calculating skewness of the column

A new variable ‘valueForSkewness’ is initialized. Mean value is subtracted from all values present in ‘salary_in_usd’ column and the whole value is multiplied by power of 3 (the formula is stored in another variable ‘skewnessSummationValue’), and the calculated value is stored in the variable ‘valueForSkewness’. Then skewness is calculated using its formula where the variable ‘valueForSkewness’ is divided by the whole calculated value of the equation where 1 is subtracted from the length of the column and it is then multiplied with the standard deviation multiplied by the power of 3.

```
[184]: valueForKurtosis = 0 #initializing a new variable
       secondSection = 0 #initializing a new variable
       thirdSection = 0 #initializing a new variable

       #using for loop in 'salary_in_usd' column for calculating values required to find out the kurtosis
       for Value in salaries['salary_in_usd']:
           valueForKurtosis = (Value - mean)
           secondSection += valueForKurtosis ** 4
           thirdSection += valueForKurtosis ** 2

[186]: kurtosis = ((secondSection / lenOfSalary) / (thirdSection / lenOfSalary) ** 2) - 3 #using the formula to calculate kurtosis

[188]: kurtosis #printing the calculated value

[188]: 0.8312989014514449
```

Figure 21: calculating kurtosis value of the column

First, mean value is subtracted from every value present in the ‘salary_in_usd’ column. Then, the calculated value is multiplied by the power of 4 and stored in ‘secondSection’ variable. Likewise, the calculated value is multiplied by the power of 2 and stored in the variable ‘thirdSection’. After that, the formula for calculating the kurtosis value is used where ‘secondSection’ variable is divided by the length of the column and the whole value is divided by the value calculated by dividing the variable ‘thirdSection’ by length of the column and multiplying the whole value by power of 2. Then, 3 is subtracted from the above calculated value. Lastly, the calculated kurtosis value is displayed.

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

```
[226]: numericColumns = salaries.select_dtypes(exclude=['object']).columns #excluding the columns containing string values

       numericColumns #displaying the columns stored in 'numericColumns'

[226]: Index(['work_year', 'salary_in_usd', 'remote_ratio'], dtype='object')
```

Figure 22: excluding columns with String values

The columns containing string values are excluded and remaining columns with numeric columns are stored in the 'numericColumns' variable. Then, the stored columns are displayed.

```
[228]: salaries[numericColumns].corr() #finding out the correlation of the numeric columns
```

[228]:

	work_year	salary_in_usd	remote_ratio
work_year	1.00000	0.228290	-0.236430
salary_in_usd	0.22829	1.000000	-0.064171
remote_ratio	-0.23643	-0.064171	1.000000

Figure 23: finding correlation of the columns

Correlation is calculated of the numeric columns using '.corr' function.

4. Data Exploration

4.1 Write a python program to find out top 15 jobs.

```
[293]: topJobs = salaries['job_title'].value_counts() #using '.value_count' function to find out the top jobs

[295]: topJobs.head(15) #printing out top 15 jobs using '.head' function

[295]: job_title
       Data Engineer                1040
       Data Scientist                840
       Data Analyst                  612
       Machine Learning Engineer     289
       Analytics Engineer            103
       Data Architect                101
       Research Scientist             82
       Data Science Manager           58
       Applied Scientist              58
       Research Engineer              37
       ML Engineer                    34
       Data Manager                   29
       Machine Learning Scientist     26
       Data Science Consultant        24
       Data Analytics Manager         22
       Name: count, dtype: int64
```

Figure 24: finding top 15 jobs

Here, the top jobs are found by using the ‘value_counts’ function which counts the number of occurrences of a certain value in ‘job_title’ column. Then, ‘.head’ function is used with 15 as its parameter to only display the top 15 jobs present in the DataFrame.

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

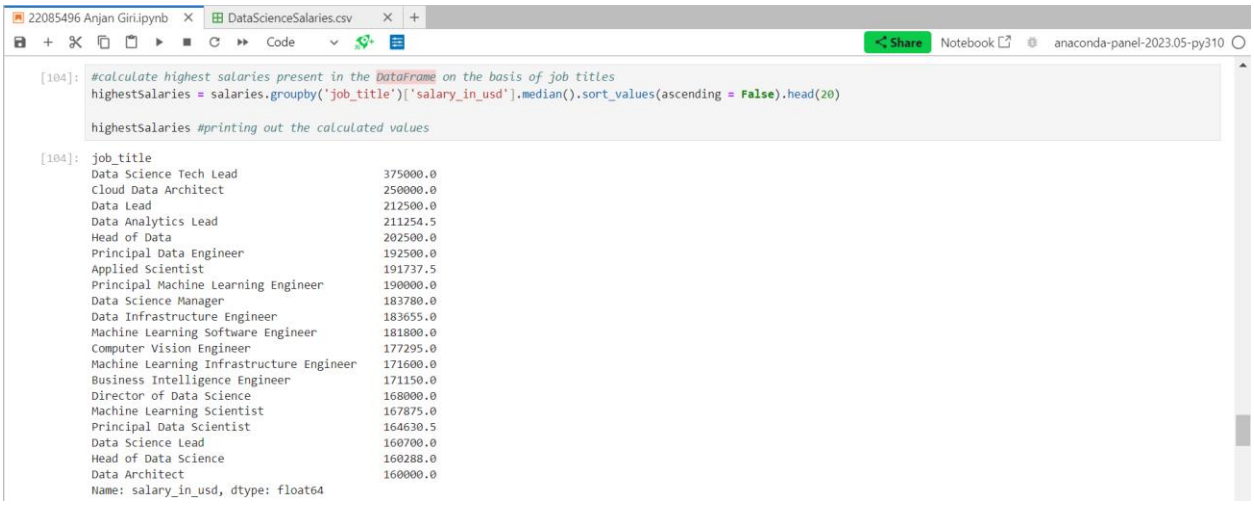


Figure 25: calculating highest salaries

The highest salaries are calculated based on job titles. The '.groupby' function is used to group the data according to the column 'job_title', '.median' is used to calculate the median of the acuumulated data and '.sort_values' is used to arrange the calculated values in descending order to get the highest values on top. After that, the '.head' function with 20 as its parameter is used to get the top 20 highest calculated salaries based on the job titles.

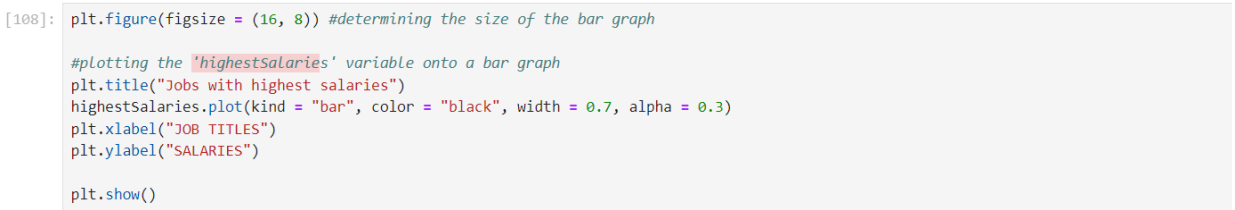


Figure 26: plotting the calculated values onto a bar graph

The highest salaries calculated on Figure: 21 are now plotted in the bar graph. The '.figure' function with 'figsize' parameter helps to determine the size of the bar graph making it easier to understand and analyze. '.title' is used to set the title of the graph and '.plot' is

used to plot the variable ‘highestSalaries’ into the graph with different parameters where ‘kind’ determines the kind of graph the values are plotted onto, ‘color’ determines the color of the bar, ‘width’ determines the space between different bars and ‘alpha’ controls the transparency of the bars.

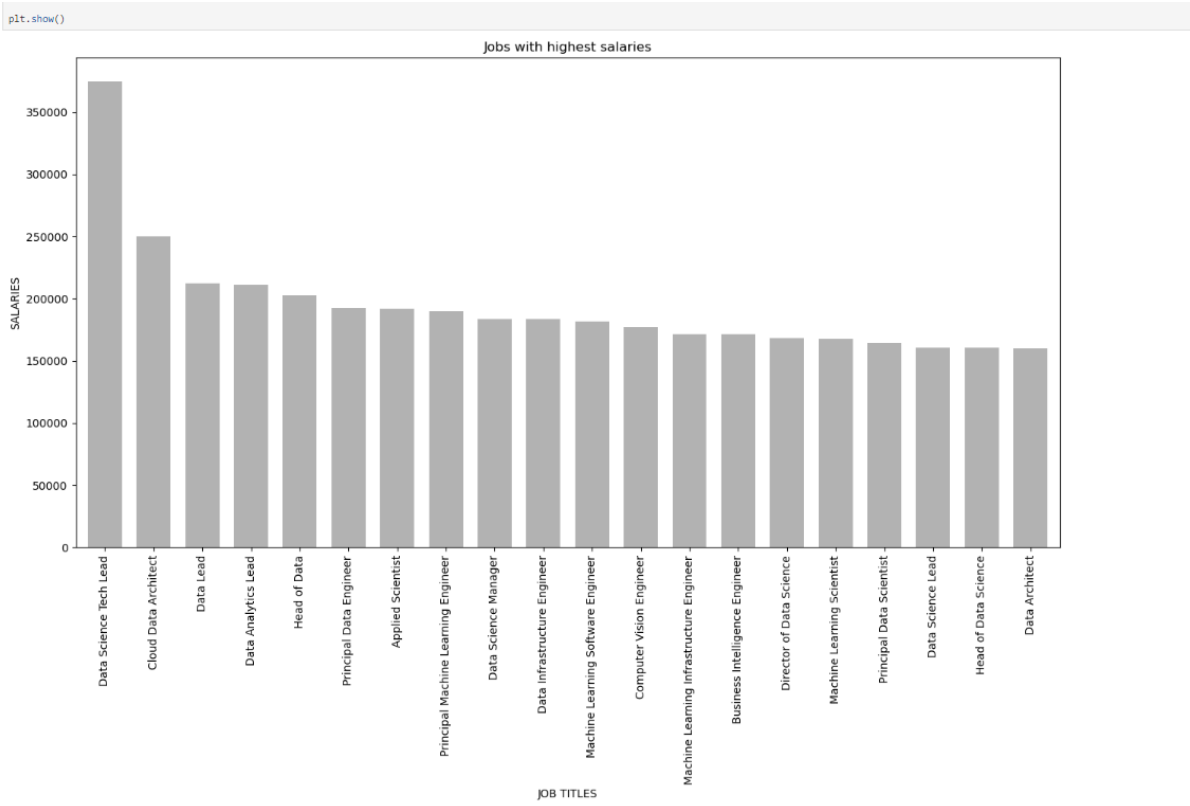


Figure 27: bar graph of jobs with highest salaries

The data is plotted and presented in the bar graph in Figure: 23 where we can see that Data Science Tech Lead has highest salaries followed by Cloud Data Architect, Data Lead and so on.

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

```
[126]: #calculating salaries based on the experience levels
       experienceSalary = salaries.groupby('experience_level')['salary_in_usd'].median()

       experienceSalary #printing out the calculated values
```

```
[126]: experience_level
       Entry Level                  70000.0
       Executive Level             196000.0
       Medium Level/Intermediate   100000.0
       Senior Level/Expert         146000.0
       Name: salary_in_usd, dtype: float64
```

Figure 28: calculating salaries based on experience level

The salaries based on experience level are calculated in the above code. The '.groupby' function is used to group the data based on the experience levels and the salaries are calculated using '.median' function in the 'salary_in_usd' column. After that the calculated salaries are printed out.

```
[120]: #plotting the 'experienceSalary' variable onto a bar graph
       plt.figure(figsize = (13, 5))

       plt.title("Salaries Based on Experience Level")
       experienceSalary.plot(kind = "bar", color = "blue", width = 0.4, alpha = 0.2)
       plt.xlabel("EXPERIENCE LEVELS")
       plt.ylabel("SALARIES")

       plt.show()
```

Figure 29: plotting the calculated salaries onto a bar graph

The calculated values of Figure: 24 are plotted onto the bar graph. The '.figure' function with 'figsize' parameter helps to determine the size of the bar graph making it easier to understand and analyze. '.title' is used to set the title of the graph and '.plot' is used to plot the variable 'experienceSalary' into the graph with different parameters where 'kind' determines the kind of graph the values are plotted onto, 'color' determines the color of the bar, 'width' determines the space between different bars and 'alpha' controls the transparency of the bars.

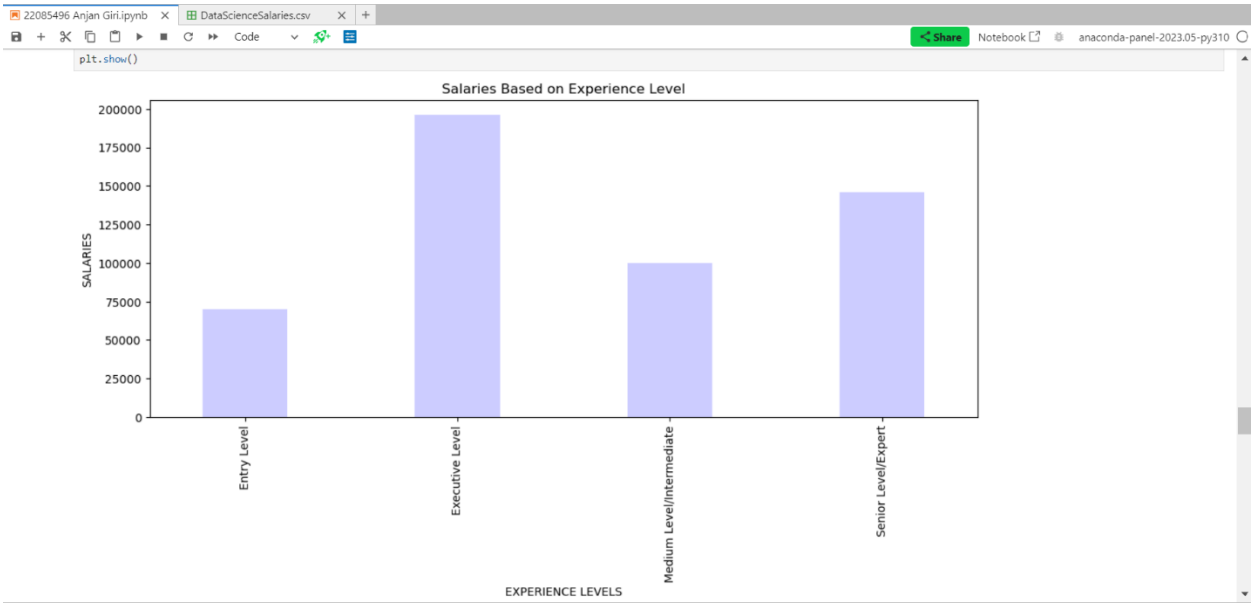


Figure 30: bar graph of salaries based on experience levels

Here, the data is plotted onto the bar graph and we can observe that the experience level with the highest salary is the Executive Level followed by Senior Level and Medium Level where the lowest salary being of Entry Level.

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

Histograms

22085496 Anjan Giri.ipynb | DataScienceSalaries.csv

Code

```
[174]: #plotting the salary_in_usd column in a histogram
plt.hist(salaries['salary_in_usd'], bins = 12, edgecolor = "red", linewidth = 0.6)

plt.title("Histogram of salary_in_usd column") #setting the title of the histogram
plt.xlabel("Salaries") #setting the x axis name

[174]: Text(0.5, 0, 'Salaries')
```

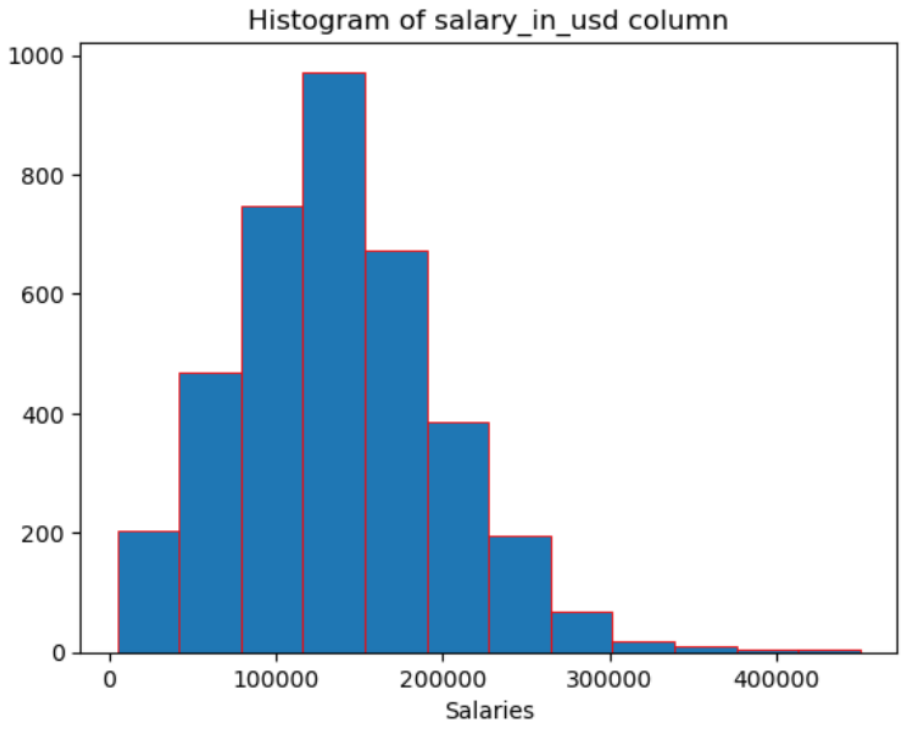


Figure 31: histogram of salary_in_usd column

The 'salary_in_usd' column is plotted into a histogram using '.hist' function. Different parameters are used in the function where 'bins' determines the number of intervals into which the data is divided into, 'edgecolor' defines the color of the edge of each bar and 'linewidth' determines the width of the border. Proper title and name for the x axis are also determined in the code.

22085496 Anjan Giri.ipynb DataScienceSalaries.csv

Code

```
[136]: #plotting the salary_in_usd column in a histogram
plt.hist(salaries['work_year'], bins = 7, edgecolor = "black", linewidth = 0.4)

plt.title("Histogram of work_year column") #setting the title of the histogram
plt.xlabel("Years") #setting the x axis name
```

```
[136]: Text(0.5, 0, 'Years')
```

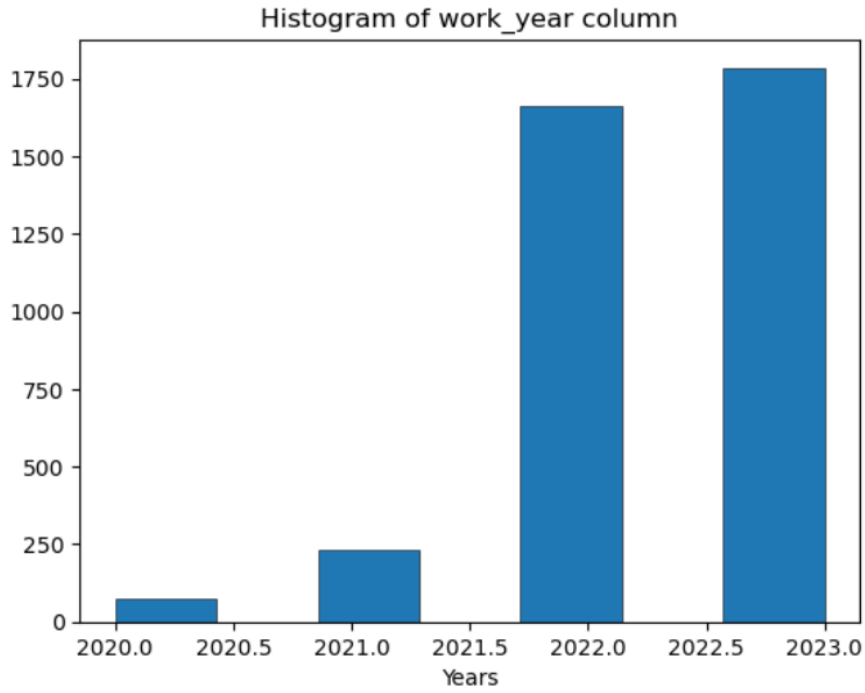


Figure 32: histogram of work_year column

The 'work_year' column is plotted into a histogram using '.hist' function. Different parameters are used in the function where 'bins' determines the number of intervals into which the data is divided into, 'edgecolor' defines the color of the edge of each bar and 'linewidth' determines the width of the border. Proper title and name for the x axis are also determined in the code.

Boxplots

```
[188]: plt.figure(figsize = (14,6)) #setting the size of the figure to make it understandable

       plt.boxplot(salaries['salary_in_usd']) #plotting the salary_in_usd column using '.boxplot'

       plt.title("Boxplot of salary_in_usd Column") #setting the title of the boxplot
```

Figure 33: plotting the column salary_in_usd into a boxplot

The ‘salary_in_usd’ column is plotted onto a boxplot using ‘.boxplot’ function. The ‘.figure’ function with figsize as its parameter gives the size of the boxplot figure and ‘.title’ function is used to set the proper title for the boxplot.

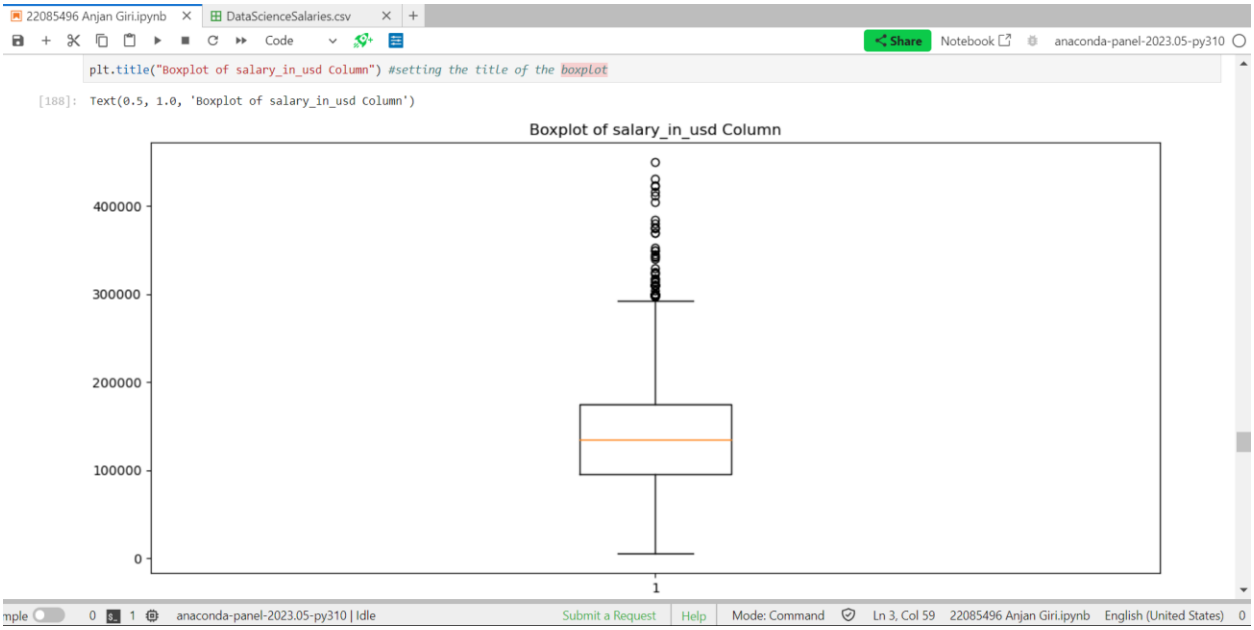


Figure 34: boxplot of salary_in_usd column

The ‘salary_in_usd’ column is plotted onto the boxplot.

```
[190]: plt.figure(figsize = (11,4)) #setting the size of the figure to make it understandable
       plt.boxplot(salaries['work_year']) #plotting the work_year column using '.boxplot'
       plt.title("Boxplot of work_year Column") #setting the title of the boxplot
```

Figure 35: plotting the column work_year into a boxplot

The 'work_year' column is plotted onto a boxplot using '.boxplot' function. The '.figure' function with figsize as its parameter gives the size of the boxplot figure and '.title' function is used to set the proper title for the boxplot

```
[190]: Text(0.5, 1.0, 'Boxplot of work_year Column')
```

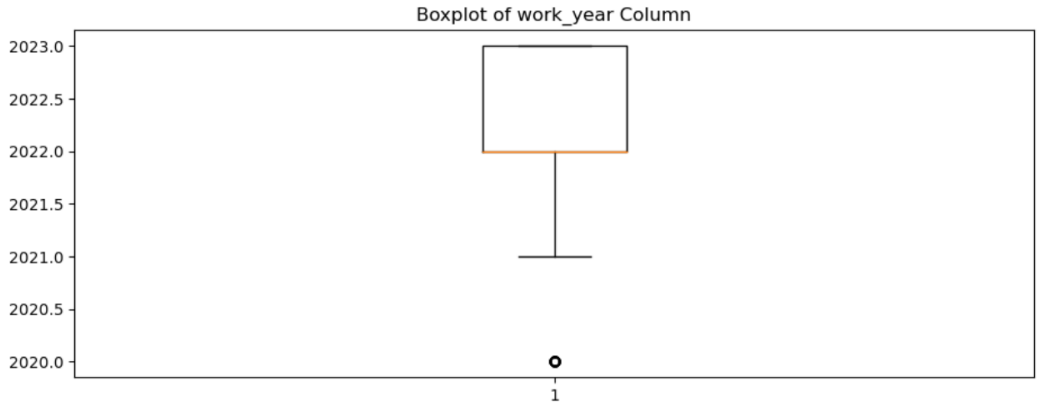


Figure 36: boxplot of work_year column

The 'work_year' column is plotted onto a boxplot.

5. Conclusion

The provided coursework of Smart Data Discovery (Module Code: CC5067NI) accounts for 60% of the total module grades. The data related to salaries of Data Science employees is provided in the question with which the coursework required us to conduct proper data analysis of the provided data and perform various tasks assigned in the question.

Upon completion of the coursework, I learnt different aspects of data understanding and analysis. The coursework helped me gain proper understanding of various topics such as data understanding, data preparation, data analysis, data exploration and document organization. The work with the provided data included retrieving required data, plotting the data and information in different kinds of graphs along with managing the whole data.

Along with the above-mentioned tasks, the coursework also covered other topics which are NumPy, pandas and matplotlib in python where NumPy is a scientific computing library, pandas is used for data analysis and manipulation and matplotlib is used for data visualization. The coursework required use of all these libraries with varieties of functions included with them.

In general, completion of the coursework provided good understanding of data analysis and manipulation using python libraries. It helped us learn different aspects of data analysis along with its visualization.