LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

CC5067NI-Smart Data Discovery

60% Individual Coursework

2023-24 Autumn

Student Name: Atal Gyawali

London Met ID: 22067674

College ID: np01cp4a220090

Assignment Due Date: Monday, May 13, 2024

Assignment Submission Date: Sunday, May 12, 2024

Word Count: 1844

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

Table of Figure

Introduction

This is the final report of the individual assessment weighted 60% of the marks for the module Smart Data Discovery. It is primarily an exercise in applying programming knowledge and skills to data analysis tasks, demonstrating our skills for problem-solving and critical thinking/evaluation. This assignment involves the Data Science salary analysis. We are expected to write Python programs and technical report on data understanding, preparation, exploration, and initial analysis.

Smart Data

Smart data is data from which signals and patterns have been extracted by intelligent algorithms. It refers to data that has been processed, analyzed, and organized in a way that makes it valuable and actionable for decision-making purposes. Unlike raw data, which may be vast and unstructured, smart data has undergone various transformations to extract meaningful insights, patterns, and trends (Dix, 2018).

Data Understanding

Data understanding is the knowledge you have about data, the needs the data will satisfy, its content and location. It involves exploring and gaining insights into the dataset to inform subsequent analysis and decision-making (Ladley, 2016). The main objectives of data understanding include:

Exploring Data Structure: This involves understanding the format of the data, including the types of variables, their relationships, and how they are organized within the dataset. Analysts may examine the data schema, data dictionaries, or metadata to understand the data's structure.

Assessing Data Quality: Analysts need to evaluate the quality of the data by checking for issues such as missing values, outliers, duplicates, and inconsistencies. Data quality assessment ensures that the data is reliable and suitable for analysis.

Identifying Patterns and Trends: Analysts look for patterns, trends, and relationships within the data to uncover insights and potential opportunities. This may involve visualizing the data using charts, graphs, or statistical methods to identify patterns and correlations.

Understanding Data Context: It's essential to understand the context in which the data was collected, including the source, purpose, and limitations. Understanding the data context helps analysts interpret the findings accurately and make informed decisions.

Documenting Findings: Finally, analysts document their findings and insights from the data understanding process. This documentation serves as a reference for future analysis and ensures transparency and reproducibility in the analytical process.

Nature of the variables given in datasets:

S.No	Column Name	Description	Data type
1	work_year	This column stores the working year.	Date
2	experience_level	This column shows the experience level of the job. Like senior level, medium level, entry level and executive level.	String
3	employment_type	This column shows the type of employment like full time or part time.	String
4	job_title	This shows the name of the job	String

		like Data Scientist, Data Analyst, Research Engineer, etc.	
5	salary	This column shows the salary according to the job, experience level and employment type.	Number
6	salary_currency	This column shows the currency on which salary are given.	String
7	salary_in_usd	This shows the salary in us Dollars.	Number
8	employee_residence	This column shows where the employees live.	String
9	remote_ratio	This shows the ration of employees working from home or a location other than a central office operated by the employer.	Number
10	company_location	This shows where the company is located.	String
11	company_size	This column shows the size of the company for example large, medium or small.	String

Data Preparation

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

   Ans: To load the any data in the data frame we must first import the pandas library. Then on the second line of code I imported the .csv file and save it in a variable called ‘file’. Then I converted the file into a data frame and saved it to another variable called ‘df’.

```
# importing pandas library
import pandas as pd

# Reading the csv file and entring the path of the file
file = pd.read_csv('D:/Smart Data (Python)/DataScienceSalaries.csv')

# Converting the file into dataframe
df = pd.DataFrame(file)
```

Figure 1: Loading data into Data Frame.

Then, to check the data frame I called the ‘df’ variable. Here, we can see that the data is loaded in the data base.

```
#display Dataframe
df
```

	work_year	experience_level	employment_type	job_title	salary	salary_currency	salary_in_usd	employee_residence	remote_ratio	company_location	c
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	
...	...	...	...	...	...	...	...	...	...	...	
3750	2020	SE	FT	Data Scientist	412000	USD	412000	US	100	US	
3751	2021	MI	FT	Principal Data Scientist	151000	USD	151000	US	100	US	
3752	2020	EN	FT	Data Scientist	105000	USD	105000	US	100	US	
3753	2020	EN	CT	Business Data Analyst	100000	USD	100000	US	100	US	
3754	2021	SE	FT	Data Science Manager	7000000	INR	94665	IN	50	IN	

3755 rows × 11 columns

Figure 2: Checking the data frame.

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

Ans: To drop the columns I used the drop keyword. The ‘inplace = True’ keyword is written to make drop the columns permanently.

```
: # Deleting the salary and salary currency columns from the dataframe
  df.drop(columns=['salary', 'salary_currency'], inplace= True)
```

Figure 3: Removing the columns.

To see if the columns are deleted or not, I called the ‘df’ variable again. Here, we can see that the two columns ‘salary’ and ‘salary_currency’ are removed.

```
# Displaying the Data Frame again to check if the colums are deleted or not
df
```

	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
3751	2021	Medium Level/Intermediate	FT	Principal Data Scientist	151000	US	100	US	L
3752	2020	Entry Level	FT	Data Scientist	105000	US	100	US	S
3753	2020	Entry Level	CT	Business Data Analyst	100000	US	100	US	L
3754	2021	Senior Level/Expert	FT	Data Science Manager	94665	IN	50	IN	L

3755 rows × 9 columns

Figure 4: Data Frame after removing columns.

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

   Ans: First, I checked if we have any missing values in our data frame. Fortunately, we do not have any NaN missing values in our data.

```
5]: # Checking if any NaN missing are present in the columns
    df.isnull().sum()

5]: 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 5: Checking for missing values.

If we had any missing values in our data frame we can easily solve this issue by ether using ‘dropna()’ command which deletes the row or using the ‘fillna()’ command from which we can enter a custom data into the missing value place.

```
#Removing NaN empty Values.
updated_df = df.dropna()

# Filling NaN values
updated_df = df.fillna(values)
```

Figure 6: Command to solving missing values.

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

Ans: To check the duplicate values in every column, I first used for loop to check data inside all the columns. Then I stored the duplicated values in a variable named ‘duplicate_values’. Then after that I converted the duplicate data into data frame. At last, I printed the duplicate values.

```
# Using for loop to access data from all of the columns
for columns in df:

    # Storing the duplicated values in a variable named duplicate_values
    duplicate_values = df[columns].duplicated()

    #framing the duplicate
    duplicate_values = df[duplicate_values]

    #Showing the duplicate values
    print(duplicate_values)
```

Figure 7: Finding duplicated values.

After printing, we can see the duplicated values in every column.

```
    #Showing the duplicate values
    print(duplicate_values)
```

```
      work_year experience_level employment_type                job_title  \
1          2023               MI              CT              ML Engineer
2          2023               MI              CT              ML Engineer
3          2023               SE              FT           Data Scientist
4          2023               SE              FT           Data Scientist
5          2023               SE              FT        Applied Scientist
...         ...              ...             ...                      ...
3750       2020               SE              FT           Data Scientist
3751       2021               MI              FT  Principal Data Scientist
3752       2020               EN              FT           Data Scientist
3753       2020               EN              CT     Business Data Analyst
3754       2021               SE              FT      Data Science Manager

       salary salary_currency  salary_in_usd employee_residence  remote_ratio  \
1       30000             USD          30000                 US           100
2       25500             USD          25500                 US           100
3      175000             USD         175000                 CA           100
4      120000             USD         120000                 CA           100
5      222200             USD         222200                 US             0
...       ...             ...            ...                ...           ...
3750   412000             USD         412000                 US           100
3751   151000             USD         151000                 US           100
3752   105000             USD         105000                 US           100
3753   100000             USD         100000                 US           100
3754  7000000             INR          94665                 IN            50
```

Figure 8: Printing duplicated values.

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

Ans: To check the unique values in every column, I first used for loop to check data inside all the columns. Then I stored the unique data in a variable and printed the column name and unique value.

```
#using for loop to select data from all of the columns
for columns in df:

    # Storing unique values in a variable
    unique_values = df[columns].unique()

    # Showing the unique values with their column names
    print(columns,unique_values)
```

Figure 9: Finding Unique values.

After printing, we can see the unique values in every column with the column name.

```
    # Showing the unique values with their column names
    print(columns,unique_values)
```

```
work_year [2023 2022 2020 2021]
experience_level ['SE' 'MI' 'EN' 'EX']
employment_type ['FT' 'CT' 'FL' 'PT']
job_title ['Principal Data Scientist' 'ML Engineer' 'Data Scientist'
 'Applied Scientist' 'Data Analyst' 'Data Modeler' 'Research Engineer'
 'Analytics Engineer' 'Business Intelligence Engineer'
 'Machine Learning Engineer' 'Data Strategist' 'Data Engineer'
 'Computer Vision Engineer' 'Data Quality Analyst'
 'Compliance Data Analyst' 'Data Architect'
 'Applied Machine Learning Engineer' 'AI Developer' 'Research Scientist'
 'Data Analytics Manager' 'Business Data Analyst' 'Applied Data Scientist'
 'Staff Data Analyst' 'ETL Engineer' 'Data DevOps Engineer' 'Head of Data'
 'Data Science Manager' 'Data Manager' 'Machine Learning Researcher'
 'Big Data Engineer' 'Data Specialist' 'Lead Data Analyst'
 'BI Data Engineer' 'Director of Data Science'
 'Machine Learning Scientist' 'MLOps Engineer' 'AI Scientist'
 'Autonomous Vehicle Technician' 'Applied Machine Learning Scientist'
 'Lead Data Scientist' 'Cloud Database Engineer' 'Financial Data Analyst'
 'Data Infrastructure Engineer' 'Software Data Engineer' 'AI Programmer'
 'Data Operations Engineer' 'BI Developer' 'Data Science Lead'
 'Deep Learning Researcher' 'BI Analyst' 'Data Science Consultant'
 'Data Analytics Specialist' 'Machine Learning Infrastructure Engineer'
 'BI Data Analyst' 'Head of Data Science' 'Insight Analyst'
 'Deep Learning Engineer' 'Machine Learning Software Engineer'
 'Big Data Architect' 'Product Data Analyst'
 'Computer Vision Software Engineer' 'Azure Data Engineer'
```

Figure 10: Printing unique values.

6. Rename the experience level columns as below.

SE – Senior Level/Expert

MI – Medium Level/Intermediate

EN – Entry Level

EX – Executive Level

Ans: To rename the columns, I used '.replace' keyword.

```
: # Permanantly renaming the columns by using replace
df['experience_level'] = df['experience_level'].replace({
    'SE':'Senior Level/Expert',
    'MI':'Medium Level/Intermediate',
    'EN':'Entry Level', 'EX':'Executive Level'})
```

Figure 11: Renaming the columns.

Now, I printed the data frame to see if the column names are changed or not. Here we can see that the SE has been changed to Senior Level/Expert, MI to Medium Level/Intermediate, EN to Entry Level and EX to Executive Level.

```
#Display the Data Frame
df
```

	work_year	experience_level	employment_type	job_title	salary	salary_currency	salary_in_usd	employee_residence	remote_ratio	company_location
0	2023	Senior Level/Expert	FT	Principal Data Scientist	80000	EUR	85847	ES	100	ES
1	2023	Medium Level/Intermediate	CT	ML Engineer	30000	USD	30000	US	100	US
2	2023	Medium Level/Intermediate	CT	ML Engineer	25500	USD	25500	US	100	US
3	2023	Senior Level/Expert	FT	Data Scientist	175000	USD	175000	CA	100	CA
4	2023	Senior Level/Expert	FT	Data Scientist	120000	USD	120000	CA	100	CA
...	...	...	...	...	...	...	...	...	...	...
3750	2020	Senior Level/Expert	FT	Data Scientist	412000	USD	412000	US	100	US
3751	2021	Medium Level/Intermediate	FT	Principal Data Scientist	151000	USD	151000	US	100	US
3752	2020	Entry Level	FT	Data Scientist	105000	USD	105000	US	100	US
3753	2020	Entry Level	CT	Business Data Analyst	100000	USD	100000	US	100	US
3754	2021	Senior Level/Expert	FT	Data Science Manager	7000000	INR	94665	IN	50	IN

3755 rows × 11 columns

Figure 12: Data Frame after renaming columns.

Data Analysis

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

Ans: First showing the user the available column. Then I used a while loop and asked the user to enter the name of the column. If the user enters the right column name, the summary statistics of that column is printed else a message is shown to the user.

```
]: # Showing users the abailable columns they can select from
print("Available columns for a summary statistics are (work_year, salary_in_usd and remote_ratio)")


while True:
    # Asking the user to select a column
    selected_column = input("Enter the name of the column to get a summary statistics /n the name of the columns are :" )

    if selected_column in ['work_year','salary_in_usd','remote_ratio']:
        # Storing the statistics in variables
        Total_sum = df[selected_column].sum()
        Total_mean = df[selected_column].mean()
        Standard_Deviation= df[selected_column].std()
        Skewness = df[selected_column].skew()
        Total_Kurtosis = df[selected_column].kurtosis()

        # Displaying all the statistics
        print("Sum                :",Total_sum)
        print("Mean               :",Total_mean)
        print("Standard Derivation:",Standard_Deviation)
        print("Skewness           :",Skewness)
        print("Kurtosis           :",Total_Kurtosis)
        break

    else:
        print("Please Enter the given columns only")
```

Figure 13: Calculating summary statistics.

The program asks the name of the column, here I entered the 'work_year' column.

```
        print("Please Enter the given columns only")
Enter the name of the column to get a summary statistics /n the name of the columns are :
work_year
Available columns for a summary statistics are (work_year, salary_in_usd and remote_ratio)
```

Figure 14: Entering the name of the column to get the summary statistics.

Here, we can see the summary statistics of the 'work_year' column.

```
    else:
        print("Please Enter the given columns only")
Available columns for a summary statistics are (work_year, salary_in_usd and remote_ratio)
Enter the name of the column to get a summary statistics /n the name of the columns are :work_year
Sum                : 7594013
Mean               : 2022.3736351531293
Standard Derivation: 0.6914482342671734
Skewness           : -1.0163743356777006
Kurtosis           : 1.1279653984751836
```

Figure 15: Summary Statistics.

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

Ans: First I selected the columns with the data in number only and stored it in a variable. Then I calculated the correlation of those columns and printed the correlation.

```
# Selecting columns that has only numeric values
numeric_columns = df.select_dtypes(include=['number'])

# Finding the correlation of the columns
correlation = numeric_columns.corr()

# Displaying the correlation
print(correlation)
```

Figure 16: Calculating the correlation.

Here, we can see the correlation of the columns with each other.

```
                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 17: Correlation of the columns.

Data Exploration

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

   Ans: First I imported the necessary libraries, then I selected data of salary in usd & job title column and framed it. Then I selected the data of top 15 rows and stored it in the job_title variable. After that I set the figure size and plotted it in the bar graph.

```
# importing the necessary libraries
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

# Selecting the colums
selected_columns = ['salary_in_usd', 'job_title']
selected_columns = df[selected_columns]

# Selecting the top 15 rows from the selected columns
top_jobs = selected_columns.head(15)

# Setting the size of the bar graph
plt.figure(figsize=(10,6))

# Plotting the data in the bar graph
top_jobs.plot(kind= 'bar', x = 'job_title', y = 'salary_in_usd', edgecolor= 'black')
```

Here, we can see that a bar graph of the top fifteen rows is created on the basis of their job title and salary in usd.

```
<Axes: xlabel='job_title'>
```

```
<Figure size 1000x600 with 0 Axes>
```

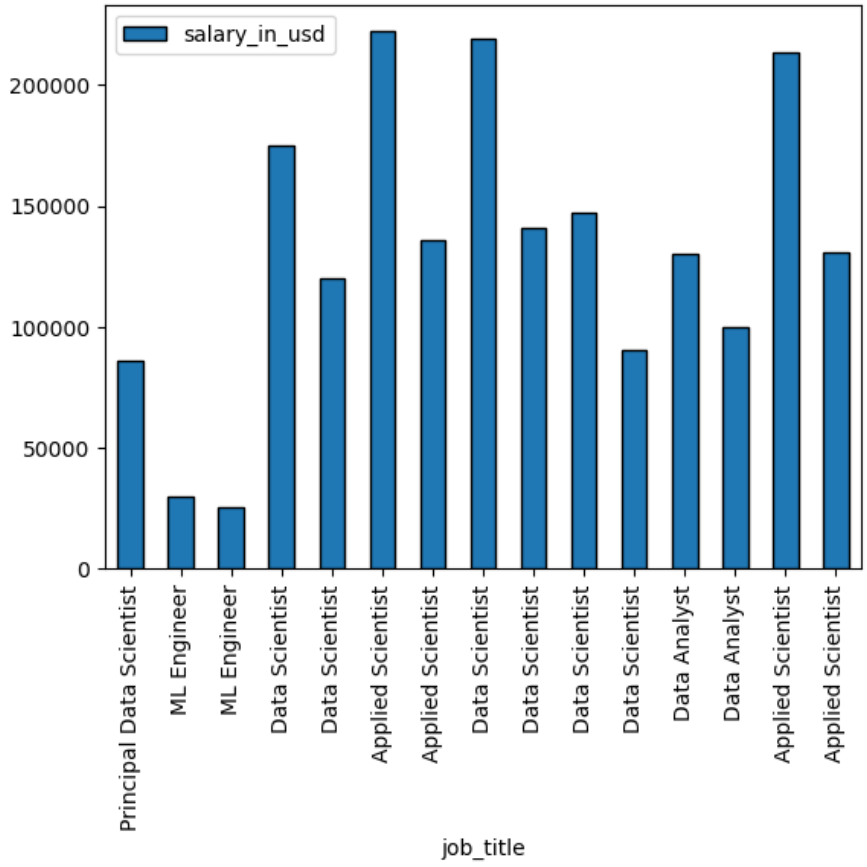

Figure 18: Bar Graph of top 15 rows.

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

Ans: First calculating the average salary of the job title, then finding the name of the job with the highest salary. Then finding the highest amount of salary. After that printing the highest job title with its average salary amount. At last, setting the figure size and plotting the data in bar graph.

```
#Calculating the average salary of each job first
Avg_salary = df.groupby('job_title')['salary_in_usd'].mean()

#Findingg the name of the job with the highest average salary
Highest_salary_job = Avg_salary.idxmax()

#Finding the highest amount of salary
Highest_salary = Avg_salary.max()

#Displaying what the highest salary job is with the amount
print("The job with the highest average salary is ",Highest_salary_job,"with the salary of ",Highest_salary)

#defining the figure size
plt.figure(figsize=(30,15))

# Plotting the info in a bar chart
Avg_salary.plot(kind = 'bar', x = 'job_title', y = 'salary_in_usd', edgecolor= 'black')
```

Figure 19: Calculating the highest salary job.

Here, we can see the bar graph of the data. We can also see that the job with the highest salary job with the salary amount is printed, which is Data Science Tech with the average salary if 375000.

```
The job with the highest average salary is  Data Science Tech Lead with the salary of  375000.0

<Axes: xlabel='job_title'>
```

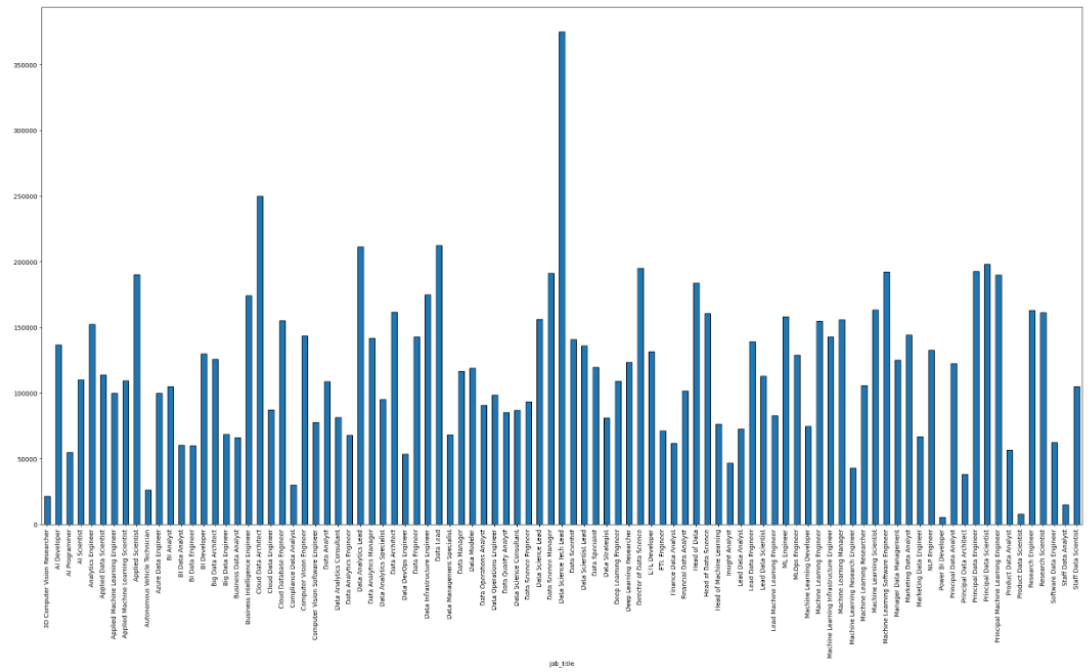

Figure 20: Bar Graph of the highest salary job.

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

Ans: First, I calculated the average based on experience level. Then I printed the amount of the salary based on experience level. Then I set the figure size and plotted the data in the bar graph.

```
# Calculating the average salary in usd based on experience level
avg_salary_per_level = df.groupby('experience_level')['salary_in_usd'].mean()

# printing the average salary in usd based on experience level
print('The average salary based on experience level are:\n',avg_salary_per_level)

#Setting the figure size
plt.figure(figsize=(10,6))

# Plotting the data in the bar graph
avg_salary_per_level.plot(kind = 'bar', x = 'experience_level', y = 'salary_in_usd', edgecolor= 'black')
```

Figure 21: Calculating the average salary based on experience level.

Here, we can see the bar graph and the average salary based on experience level.

```
The average salary based on experience level are:
 experience_level
Entry Level                  78546.284375
Executive Level             194930.929825
Medium Level/Intermediate   104525.939130
Senior Level/Expert         153051.071542
Name: salary_in_usd, dtype: float64
```

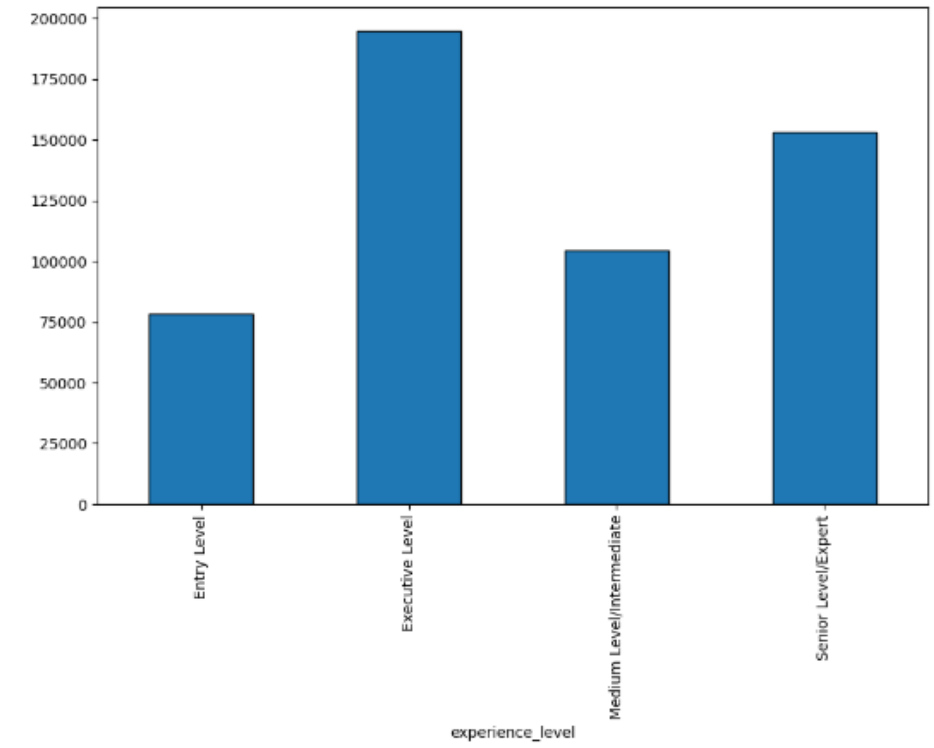

Figure 22: Bar Graph of the salary based on experience level.

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

   Ans: First I printed the names of the columns that have numerical values and asked the user to enter the name of column to get the histogram. If user enters the right column name, the program creates a labeled histogram else shows an error message.

```
]: # Showing users the abailable columns they can select from
   print("Available columns for histogram are (work_year, salary_in_usd and remote_ratio)")


   while True:
       # Asking the user to select a column
       selected_column = input("Enter the name of the column to get the histogram :" )

       # If the users enter the right column name
       if selected_column in ['work_year','salary_in_usd','remote_ratio']:

           # Creating box plot
           df.hist(column = selected_column, bins = 100,);

           # Lebeling the figure
           plt.xlabel(selected_column)
           plt.ylabel('Frequency')
           plt.title('Histogram of ' + selected_column)

           break

       # If the user enters any thing other than the right column
       else:
           print("please enter from the given column only")
```

Figure 23: Creating a histogram.

The program asks to enter the column name. Here, I entered ‘work_year’.

```
Enter the name of the column to get the histogram : work_year

Available columns for histogram are (work_year, salary_in_usd and remote_ratio)
```

Figure 24: Entering column name to get histogram.

Here, we can see the histogram of ‘work_year’ column.

```
Available columns for a summary statistics are (work_year, salary_in_usd and remote_ratio)
Enter the name of the column to get a summary statistics /n the name of the columns are :work_year
```

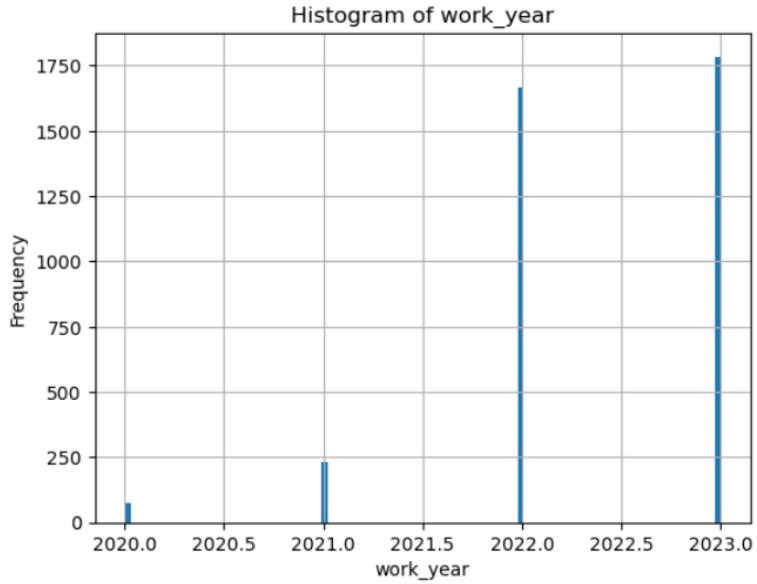

Figure 25: Histogram of the work year column.

Here, we can see the histogram of ‘salary_in_usd’ column.

```
Available columns for a summary statistics are (work_year, salary_in_usd and remote_ratio)
Enter the name of the column to get a summary statistics /n the name of the columns are :salary_in_usd
```

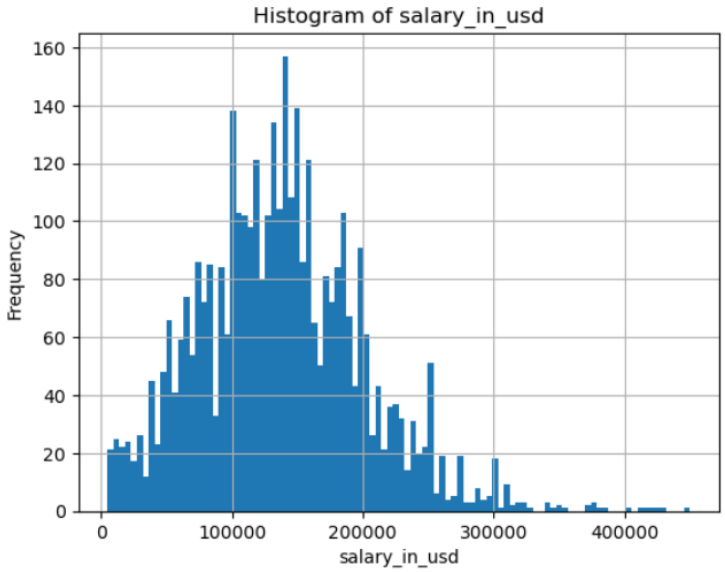

Figure 26: Histogram of the salary in usd column.

Here, we can see the histogram of the ‘remote_ratio’ column.

```
Available columns for a summary statistics are (work_year, salary_in_usd and remote_ratio)
Enter the name of the column to get a summary statistics /n the name of the columns are :remote_ratio
```

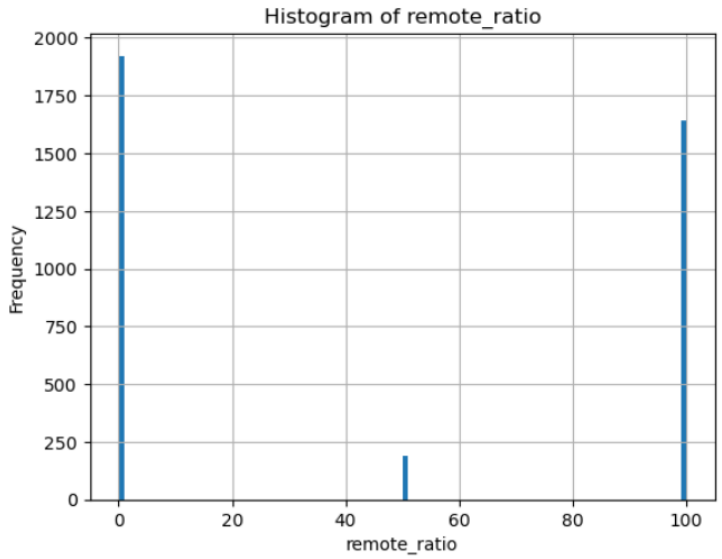

Figure 27: Histogram of the remote ratio column.

Now, for the box plot I printed the names of the columns that have numerical values and asked the user to enter the name of column to get the box plot. If user enters the right column name, the program creates a labeled box plot else shows an error message.

```
# Showing users the abailable columns they can select from
print("Available columns for a box plot are (work_year, salary_in_usd and remote_ratio)")


while True:
    # Asking the user to select a column
    selected_column = input("Enter the name of the column to get the box plot :" )

    # If the users enter the right column name
    if selected_column in ['work_year','salary_in_usd','remote_ratio']:

        # Creating box plot
        df.boxplot(column = selected_column, figsize=(10,6))

        # Labeling the figure
        plt.ylabel('Values')
        plt.title('Box Plot of ' + selected_column)
        break

    # If the user enters any thing other than the right column
    else:
        print("please enter from the given column only")
```

Figure 28: Creating box plot.

The program asks to enter the column name. Here, I entered ‘salary_in_usd’.

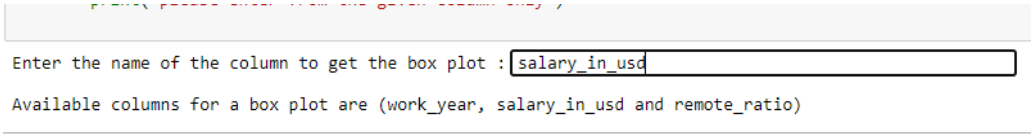

Figure 29: Entering column name to get box plot.

Here, we can see the box plot of ‘salary_in_usd’ column.

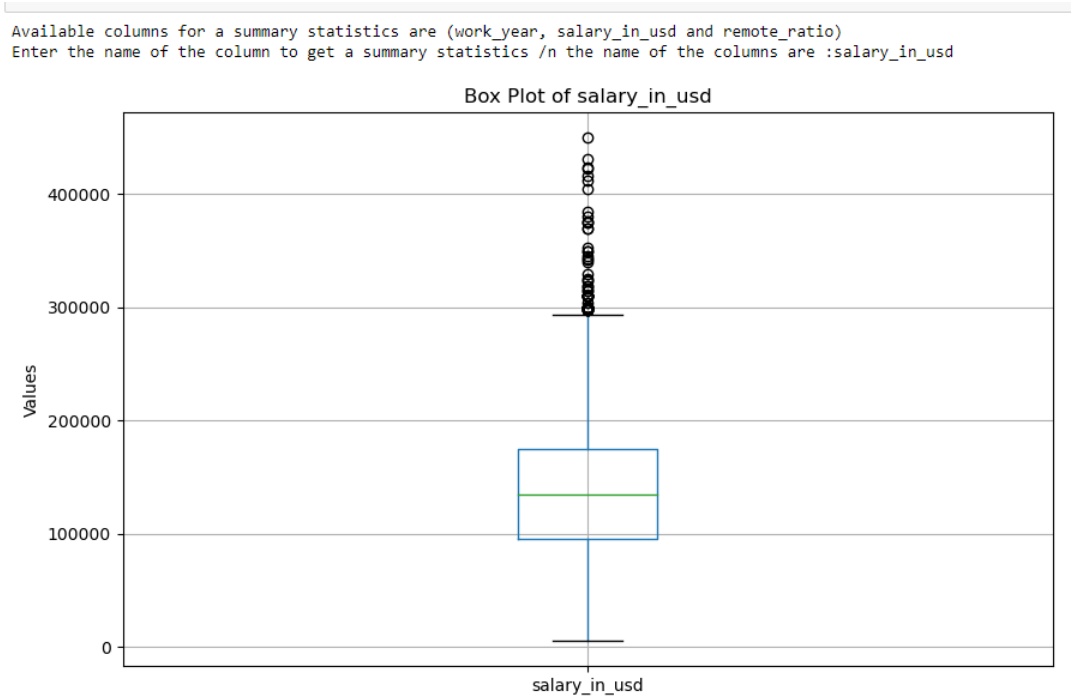

Figure 30: Box plot of the salary in usd column.

Here, we can see the box plot of ‘work_year’ column.

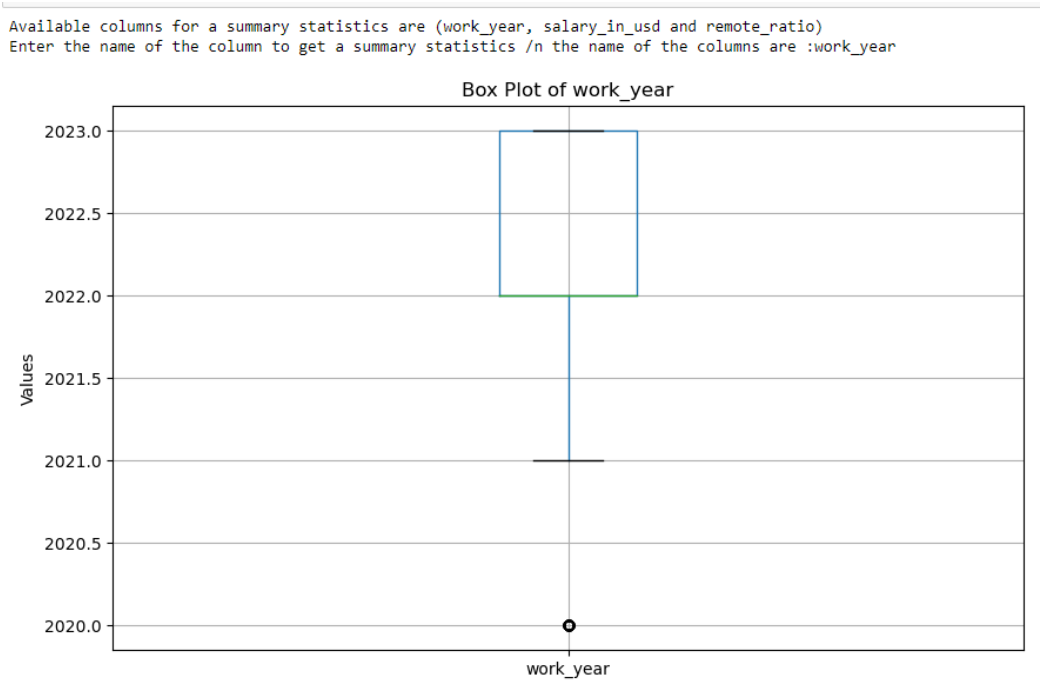

Figure 31: Box plot of the work year column.

Here, we can see the box plot of ‘remote ratio’ column.

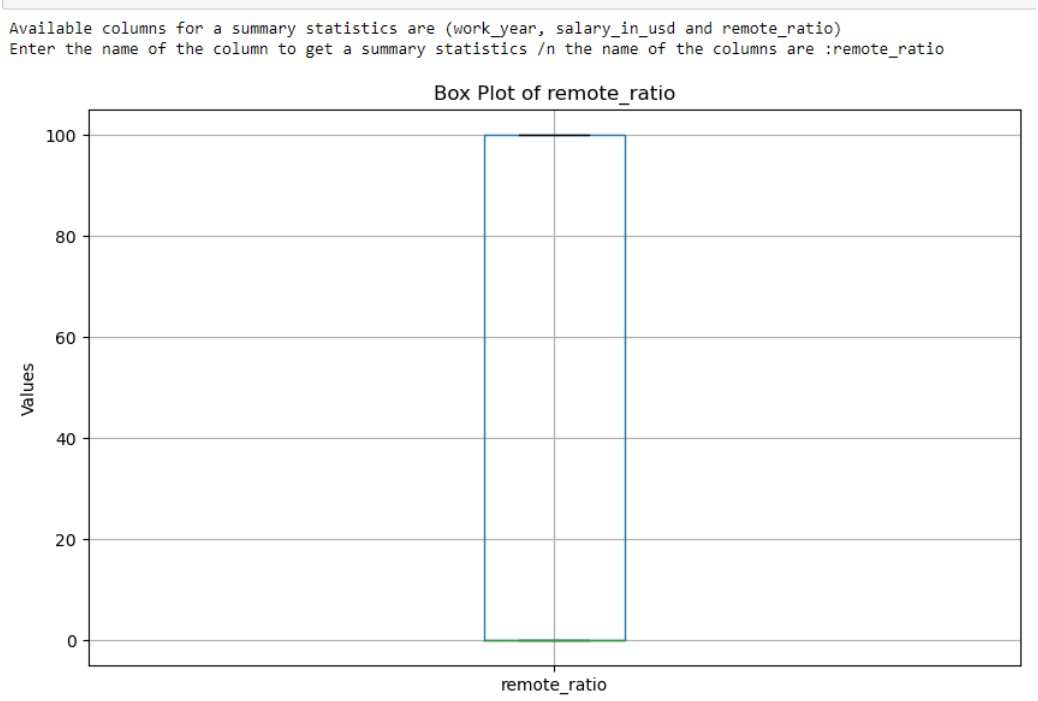

Figure 32: Box plot of the remote ration column.

Conclusion

This assessment has been a valuable learning experience for me. I've gained a better understanding of how to work with data, analyze it, and draw meaningful insights. I've learned various techniques for preparing and cleaning data, as well as how to use Python to perform statistical analysis and create visualizations. By solving different tasks in the assessment, I've improved my problem-solving skills and become more confident in using Python for data tasks. Additionally, I now understand the importance of clear documentation and reporting to communicate analysis findings effectively. Overall, this assessment has helped me grow and prepare for future data science projects.

Bibliography

Dix, J., 2018. *What is Smart Data? How Does it Help?.* [Online]
Available at: https://www.netscout.com/blog/what-smart-data-how-does-it-help
[Accessed 3 May 2024].

Ladley, J., 2016. *Mastering and managing data understanding.* [Online]
Available at: https://www.cio.com/article/238649/mastering-and-managing-data-understanding.html
[Accessed 3 May 2024].