CC5067NI-Smart Data Discovery

60% Individual Coursework

2023-24 Autumn

Student Name: Ashim Poudel

London Met ID: 22085505.

College ID: np01cp4s230145

Assignment Due Date: Monday, May 13, 2024

Assignment Submission Date: Monday, May 13, 2024

Word Count: 3015

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

Table of Contents

Table of Figures. 3

Table of Tables 3

1. Data Understanding 4

2. Data preparation 6

2.1 Write a python program to load data into pandas DataFrame. 6

2.2 Write a python program to remove unnecessary columns i.e., salary and salary currency. 7

2.3 Write a python program to remove the NaN missing values from updated dataframe. 9

2.4 Write a python program to check duplicates value in the dataframe. 10

2.5 Write a python program to see the unique values from all the columns in the dataframe. 11

2.6 Rename the experience level columns as below. 12

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

3.1.1 Sum 14

3.1.2 Mean 15

3.1.3 Standard Deviation 15

3.1.4 skewness 16

3.1.5 kurtosis 16

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

4. Data Exploration 19

4.1 Write a python program to find out top 15 jobs. Make a bar graph of sales as well. 19

4.2 Which job has the highest salaries? Illustrate with bar graph. 22

4.3 Write a python program to find out salaries based on experience level. Illustrate it through bar graph. 24

4.4 Write a Python program to show histogram and box plot of any chosen different variables. Use proper labels in the graph. 25

5. Conclusion 27

6. References 28

Table of Figures.

Figure 1 Description of Data.. 5
Figure 2 loading data into pandas DataFrame 7
Figure 3 Removing unnecessary columns. 8
Figure 4 Removing the NaN missing values from dataframe. 9
Figure 5 Checking duplicates value in the dataframe.................................... 10
Figure 6 Unique values from all columns. 11
Figure 7 Unique values from all columns. 12
Figure 8 Renaming columns. 13
Figure 9 calculating the sum 15
Figure 10 calculating the mean 15
Figure 11 calculating the standard deviation. 16
Figure 12 Calculating the skewness.. 16
Figure 13 Calculating the Kurtosis 16
Figure 14 calculating correlation of all variables... 17
Figure 15 calculating correlation of all variables continue.... 18
Figure 16 Plotting top 15 jobs in bar graph... 20
Figure 17 Top 15 jobs in bar graph. 21
Figure 18 Plotting Highest paid jobs.. 22
Figure 19 Highest paid job. 23
Figure 20 plotting Salaries based on experience level. 24
Figure 21 Histogram and box plot of any chosen different variables. 25
Figure 22 Histogram and Box plott of salary_in_usd. 26

Table of Tables

Table 1: Data understanding 5

1. Data Understanding

The dataset includes facts on individual's employment, such as job roles, pay, types of employment, and locations of employment, in addition to information about the businesses or organizations they are employed by. It could be applied to several investigations, including understanding the demographics of the organization, employment patterns, remote work preferences, and compensation distributions. The dataset consists of 11 columns in total. The table below contains descriptions of the columns that make up the dataset.

S.N	Column Name	Description	Data Type
1	work_year	This column shows the year of employment or the period the data was collected.	date
2	experience_level	This column shows the employees experience level which is denoted as SE, MI, EX, EN.	Varchar
3	employment_type	This column shows each person employment type such as FT, PT, CT.	Varchar
4	job_title	This column shows the title of jobs that the employees do in the organization.	Varchar
5	salary	This column shows the salary of the employees.	Int
6	salary_currency	The currency in which the salary is denominated is specified in this column.	Varchar
7	salary in usd	If salaries are in different currencies, this column may provide a standard representation of the salary in USD, making comparison and analysis simpler.	int
8	employee_residence	This column shows where the employee resides.	Varchar

9	remote_ratio	This column shows each employee's percentage of remote work compared to on-site work.	int
10	company_location	The companies or organizations that the employees work for are listed in this column.	varchar
11	company_size	This column, which is labeled as "S" for small, "M" for medium, or "L" for large, shows the size of the businesses or organizations where the workers are employed.	varchar

Table 1: Data understanding

The information about the data presents there can also be found within the python. For that, we can use an .info() method. This method will return the name of each column with the amount of not-null value and its datatype as shown in the figure below.

```
[87]: data_frame.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3755 entries, 0 to 3754
Data columns (total 9 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   work_year           3755 non-null   int64 
 1   experience_level    3755 non-null   object
 2   employment_type     3755 non-null   object
 3   job_title           3755 non-null   object
 4   salary_in_usd       3755 non-null   int64 
 5   employee_residence  3755 non-null   object
 6   remote_ratio        3755 non-null   int64 
 7   company_location    3755 non-null   object
 8   company_size        3755 non-null   object
dtypes: int64(3), object(6)
memory usage: 264.2+ KB
```

Figure 1 Description of Data

2. Data preparation

Data preparation is the process of cleaning, transforming, and organizing data to make it ready for analysis. You transform your data from its raw form to an appropriate format to prepare it for visualization or modelling. Data preparation includes several steps: handling missing and outlying values, detecting inconsistencies, transforming variables, integrating multiple sources, reducing dimensionality, formatting for the appropriate type, and splitting the data. Properly, by preparing your data in the right way, you as an analyst can be sure that the information is accurate, integral, and ready for further exploration or modelling.

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

Using the pd.read_csv() function to load data on the Pandas DataFrame. Other functions for loading various types of data are pd.read_excel() for Excel files, pd.read_sql() for reading SQL, etc. These functions do not just read, but also automatically load the data into a DataFrame. DataFrame is a two-dimensional labeled data structure with columns of potentially different types. Must also specify the dataset's file path or URL. Moreover, you can use other parameters to adjust the loading process, for example, specify the column names, the index column, data types, and determine how to handle missing values. We can perform a variety of data analysis and data manipulation tasks with a Pandas DataFrame using corresponding functions and methods.

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

[2]:
```
#   Write a python program to load data into pandas DataFrame
data_frame = pd.read_csv('dataset.csv')
data_frame.head()
```

[2]:

ear	experience_level	employment_type	job_title	salary	salary_currency	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
)23	SE	FT	Principal Data Scientist	80000	EUR	85847	ES	100	ES	L
)23	MI	CT	ML Engineer	30000	USD	30000	US	100	US	S
)23	MI	CT	ML Engineer	25500	USD	25500	US	100	US	S
)23	SE	FT	Data Scientist	175000	USD	175000	CA	100	CA	M
)23	SE	FT	Data Scientist	120000	USD	120000	CA	100	CA	M

Figure 2 loading data into pandas DataFrame

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

This Python program helps eliminate two columns, ‘salary’ and ‘salary_currency’, that are not important from the DataFrame called ‘data_frame’. To accomplish this operation, I applied the drop function of the Pandas library and specified the columns that I wanted to remove using the column axis, which is the newly created DataFrame without the specified columns was displayed. The given code represents a common operation of preparing some data for analysis or further modelling. By removing irrelevant or uninformative data, in this case, dataset is clean and cleaner and make it more streamlined or neat.

```
[16]: #  Write a python program to remove unnecessary columns i.e., salary and salary currency.
data_frame = data_frame.drop(['salary', 'salary_currency'], axis=1)
data_frame
```

[16]:

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_residence	remote_ratio	com
0	2023	SE	FT	Principal Data Scientist	85847	ES	100	
1	2023	MI	CT	ML Engineer	30000	US	100	
2	2023	MI	CT	ML Engineer	25500	US	100	
3	2023	SE	FT	Data Scientist	175000	CA	100	
4	2023	SE	FT	Data Scientist	120000	CA	100	
...	...	...	...	...	...	...	...	
3750	2020	SE	FT	Data Scientist	412000	US	100	
3751	2021	MI	FT	Principal Data Scientist	151000	US	100	
3752	2020	EN	FT	Data Scientist	105000	US	100	
3753	2020	EN	CT	Business Data Analyst	100000	US	100	
3754	2021	SE	FT	Data Science Manager	94665	IN	50	

3755 rows × 9 columns

Figure 3 Removing unnecessary columns.

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

The columns ‘salary’ and ‘salary_currency’ in the DataFrame ‘data_frame’ are deleted using the drop method of the Pandas library. It is used so that irrelevant columns are not present in the DataFrame and when the data relies on columns containing only relevant information may be analysed or processed. Next, the DataFrame obtained without the columns under consideration is displayed. Such a code snippet is a vivid example of standard data manipulation tool to simplify the quality and productive characteristics of the subsequent data processing procedures. Both columns contain information that becomes irrelevant and should be kept out to simplify the operation with such type of data.

[20]:
```
# Write a python program to remove the NaN missing values from updated dataframe.
data_frame.isnull()
```

[20]:

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

3755 rows × 9 columns

Figure 4 Removing the NaN missing values from dataframe.

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

The Python program checks for duplicate rows in the DataFrame by using the duplicated() method. It generates a boolean Series denoting duplicate rows with respect to their previous occurrences. Therefore, calling data_frame.duplicated() identifies duplicate rows in the DataFrame, and this could be used to ensure data quality and detect anomalies such as duplication of data.

```
[23]: # Write a python program to check duplicates value in the dataframe.
      data_frame.duplicated()
```

```
[23]: 0       False
      1       False
      2       False
      3       False
      4       False
              ...
      3750    False
      3751    False
      3752    False
      3753    False
      3754    False
      Length: 3755, dtype: bool
```

Figure 5 Checking duplicates value in the dataframe

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

This program iterates through all columns of the DataFrame and prints out the unique values for each column. Using each column by data_frame[column] and by applying the function unique() identifies the unique values that are present in each column for display. This can be a fast way to get an overview of the unique values across all columns in a DataFrame for exploration and understanding purposes.

```
[33]: # Write a python program to see the unique values from all the columns in the dataframe.
for column in data_frame.columns:
    unique_value = data_frame[column].unique()
    print(f"unique value in column '{column}':")
    print(unique_value)
    print()
```

```
unique value in column 'work_year':
[2023 2022 2020 2021]

unique value in column 'experience_level':
['SE' 'MI' 'EN' 'EX']

unique value in column 'employment_type':
['FT' 'CT' 'FL' 'PT']

unique value in column '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'
```

Figure 6 Unique values from all columns.

```
unique value in column 'salary_in_usd':
[ 85847  30000  25500 ...  28369 412000  94665]

unique value in column '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 column 'remote_ratio':
[100   0  50]

unique value in column '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 column 'company_size':
['L' 'S' 'M']
```

Figure 7 Unique values from all columns.

2.6 Rename the experience level columns as below.

SE – Senior Level/Expert

MI – Medium Level/Intermediate

EN – Entry Level

EX – Executive Level

This program renames specific columns from the DataFrame by a provided mapping. It updates the column names using the rename() method with a dictionary where the old column names are keys and the new names are values. This allows giving names to the columns that are clear and self-descriptive for readability and better understanding of the data.

[39]:
```
'''
Rename the experience level columns as below.
SE - Senior Level/Expert
MI - Medium Level/Intermediate
EN - Entry Level
EX - Executive Level
'''
data_frame = data_frame.rename(columns={'SE': 'Senior Level/Expert', 'MI': 'Medium Level/Intermediate',
                                        'EN': 'Entry Level', 'EX': 'Executive Level'})
data_frame
```

[39]:

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_residence	remote_ratio	com
0	2023	SE	FT	Principal Data Scientist	85847	ES	100	
1	2023	MI	CT	ML Engineer	30000	US	100	
2	2023	MI	CT	ML Engineer	25500	US	100	
3	2023	SE	FT	Data Scientist	175000	CA	100	
4	2023	SE	FT	Data Scientist	120000	CA	100	
...	...	...	...	...	...	...	...	
3750	2020	SE	FT	Data Scientist	412000	US	100	
3751	2021	MI	FT	Principal Data Scientist	151000	US	100	
3752	2020	EN	FT	Data Scientist	105000	US	100	
3753	2020	EN	CT	Business Data Analyst	100000	US	100	

Figure 8 Renaming columns.

3. Data Analysis

Data analysis is a process with many facets that includes several techniques and methods allowing to derive actionable data from the information. Initially, exploratory data analysis is conducted to understand the dataset structure and specifics, which is done by examining summary statistics, distributions of values and visualizations. Such analysis allows the researchers to understand which value and features are the most important, and which are the outliers that potentially require further attention. After the exploratory part, the data should be cleaned and pre-processed to fix missing values, outliers and logical inconsistencies in data thus improving the data for further analysis. Some statistical methods may be used to quantify the relationships and identify if there are any substantial connections and how these relationships can be utilized. Moreover, in most cases machine learning algorithms are included to build predictive models or investigate data patterns. At the final stage, the analysis results are interpreted and delivered to the stakeholders in the form of visual reports, which will allow them to understand their data and make use of it for business or strategic planning.

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

3.1.1 Sum

This program calculates the sum of the values in the column 'salary_in_usd' and provides summary statistics. The program goes through each value within that column, accumulating the sum of the values and then counting the total number of values. Finally, it will print out the sum and the total amount of values in the column. This will provide a basic summary statistic of the total amount of salary and the number of data points in the 'salary_in_usd' column.

```
[47]: # Write a Python program to show summary statistics of sum, mean, standard deviation,
#skewness, and kurtosis of any chosen variable.

#sum
column = data_frame['salary_in_usd']
total = 0
sum = 0

for data in column:
    total += 1
    sum+= data

print("The sum is ", sum)
print("THE TOTAL NUM IS ", total)

The sum is  516576814
THE TOTAL NUM IS  3755
```

Figure 9 calculating the sum

3.1.2 Mean

The program calculates the mean the average value of values in the 'salary_in_usd' column by division of the sum of values by the total number of values. The mean is an average amount of salary in the column.

```
[50]: mean = sum/total
print("Mean:", mean)

Mean: 137570.38988015978
```

Figure 10 calculating the mean

3.1.3 Standard Deviation

The program computes the standard deviation of values in the 'salary_in_usd' column.

```
[56]: # standard deviation
      a1_minus_mean = 0
      for data in column:
          a1_minus_mean = a1_minus_mean+(data-mean)**2
      standardDev = ((a1_minus_mean)/(total -1))**(1/2)
      a1_minus_mean

[56]: 14925948594621.205
```

Figure 11 calculating the standard deviation.

3.1.4 skewness

```
[48]: # calculating skewness.
      skewness = ((a1_mean)**3)/((total-1)*(a1_mean**3))
      skewness

[48]: 0.0002663825253063399
```

Figure 12 Calculating the skewness

3.1.5 kurtosis

```
[68]: #kurtosis
      kurtosis = ((a1_mean)*4)/((total-1)*(a1_mean**4))
      kurtosis

[68]: 3.2043496448849494e-43
```

Figure 13 Calculating the Kurtosis

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

•[14]:
```python
# Write a Python program to calculate and show correlation of all variables.
from scipy.stats import skew

user_input = input(f"Give a column name from these options: {', '.join(data_frame.columns)}\n")
if user_input in data_frame.columns:
    print("Column is present")

    col = data_frame[user_input]

    sum_ = col.sum()
    total = len(col)
    print("The sum is", sum_)
    print("The total num is", total)

    mean = sum_ / total
    print("The mean is", mean)

    a1_mean = 0
    for data in col:
        a1_mean += (data - mean) ** 2
    standardDev = ((a1_mean) / (total - 1)) ** 0.5
    print("The standard deviation is", standardDev)
    print()

    skewness = skew(col)
    print("The skewness is", skewness)

    second_column = input("Enter the name of the second column for correlation calculation:\n")

    if second_column in data_frame.columns:
        second_col = data_frame[second_column]

        a2_mean = second_col.mean()
        a2_mean = 0
        for data in second_col:
            a2_mean += (data - a2_mean) ** 2
        a2_standardDev = ((a2_mean) / (total - 1)) ** 0.5
```

Figure 14 calculating correlation of all variables

```
        covariance_sum = 0
        for a1, a2 in zip(col, second_col):
            covariance_sum += (a1 - mean) * (a2 - a2_mean)
        covariance = covariance_sum / (total - 1)

        correlation = covariance / (standardDev * a2_standardDev)

        print(f"The correlation between {user_input} and {second_column} is:", correlation)
    else:
        print("Second column doesn't exist.")
else:
    print("We do not have such column")
```

```
Give a column name from these options: work_year, experience_level, employment_type, job_title, salary_in_u
sd, employee_residence, remote_ratio, company_location, company_size
 salary_in_usd
Column is present
The sum is 516576814
The total num is 3755
The mean is 137570.38988015978
The standard deviation is 63055.625278224084

The skewness is 0.5361868674235593
Enter the name of the second column for correlation calculation:
 experience_level
```

Figure 15 calculating correlation of all variables continue....

4. Data Exploration

Data exploration is a process of deducing valuable information from a dataset for comprehensive understanding and further data analysis. The basic goal of this activity is to find patterns and characteristics of a dataset, discover connections, and syncs between its variables, and represent data in graphical form. Specifically, such exploration methods as summarization, which includes finding the characteristics of the data, data visualization, and correlation analysis are commonly used in data exploration for achieving more profound insights. The purpose of such process is to understand and uncover the dataset's structure, evaluate its important characteristics, and see whether there are any general and meaningful patterns or unexpected outliers or gaps in the data. It should be noted that such methods are used for initiating more profound analysis or a meaningful dataset model creation and may vary across different data types.

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

The program is extracting the count of each unique job title from the column 'job_title' of the DataFrame. Then, the program is identifying the top 15 most frequent job titles and producing a bar graph regarding the frequency of their occurrences. The function value_counts() calculates the frequency of each job title. Moreover, head(15) selects the top 15 job titles. The program is utilizing the Matplotlib library to create a bar graph. The x-axis is job_title, and the y-axis is the count of the job_title. Finally, the graph shows using plt.show(). This code yields a rapid graphical representation of the job_title distribution in the dataset.

```
[18]: # Write a python program to find out top 15 jobs. Make a bar graph of sales as well.
job_extract = data_frame['job_title'].value_counts()
print(job_extract)
top_15_jobs = job_extract.head(15)
print(top_15_jobs)

plt.figure(figsize=(10, 5))
top_15_jobs.plot(kind="bar", color='green')
plt.title('15 Top Job Titles')
plt.xlabel('Job Title')
plt.ylabel('count')



plt.show()
```

```
job_title
Data Engineer                1040
Data Scientist                840
Data Analyst                  612
Machine Learning Engineer     289
Analytics Engineer            103
                             ...
Compliance Data Analyst         1
Data Science Tech Lead          1
Head of Machine Learning        1
Staff Data Scientist            1
Finance Data Analyst            1
Name: count, Length: 93, dtype: int64
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
```

Figure 16 Plotting top 15 jobs in bar graph.

```
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
```

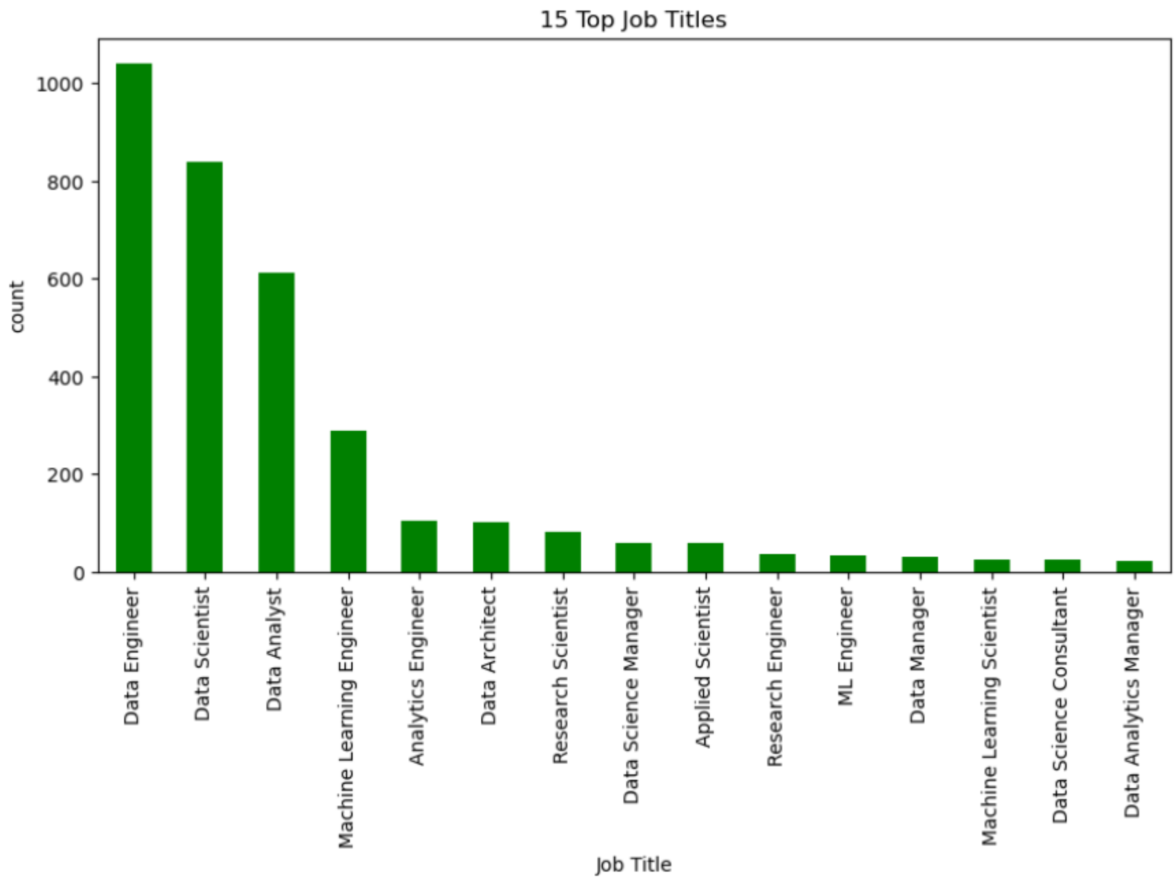

Figure 17 Top 15 jobs in bar graph.

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

The program obtains the average salary of each job title by grouping the salaries with the same job titles and calculating their mean. Then, it sorts the job titles according to their average salary in a descending order and further selects the top 10 job titles with the highest average salary. A bar graph of these selected top 10 job titles and their average salary was developed using Matplotlib. The x-axis entails the job titles as the label while the y-axis has the average salary in USD as labels. The output graph was shown using plt.show() to represent a visual for the job titles with the highest salaries.

```python
# Which job has the highest salaries? Illustrate with bar graph.
jobs = data_frame.job_title
salaries = data_frame.salary_in_usd

# Calculate average salaries for each job title
avg_salaries = salaries.groupby(jobs).mean()

# Sort job titles by average salary in descending order
sorted_jobs = avg_salaries.sort_values(ascending=False)

# Select top 10 jobs with highest average salaries
top_10_jobs = sorted_jobs.head(10)

# Plotting
plt.figure(figsize=(10, 5))
plt.bar(top_10_jobs.index, top_10_jobs.values)
plt.xlabel('Various Jobs')
plt.ylabel('Average Salary in USD')
plt.title('Jobs with Highest Salaries')
plt.xticks(rotation=90)
plt.show()
```

Figure 18 Plotting Highest paid jobs.

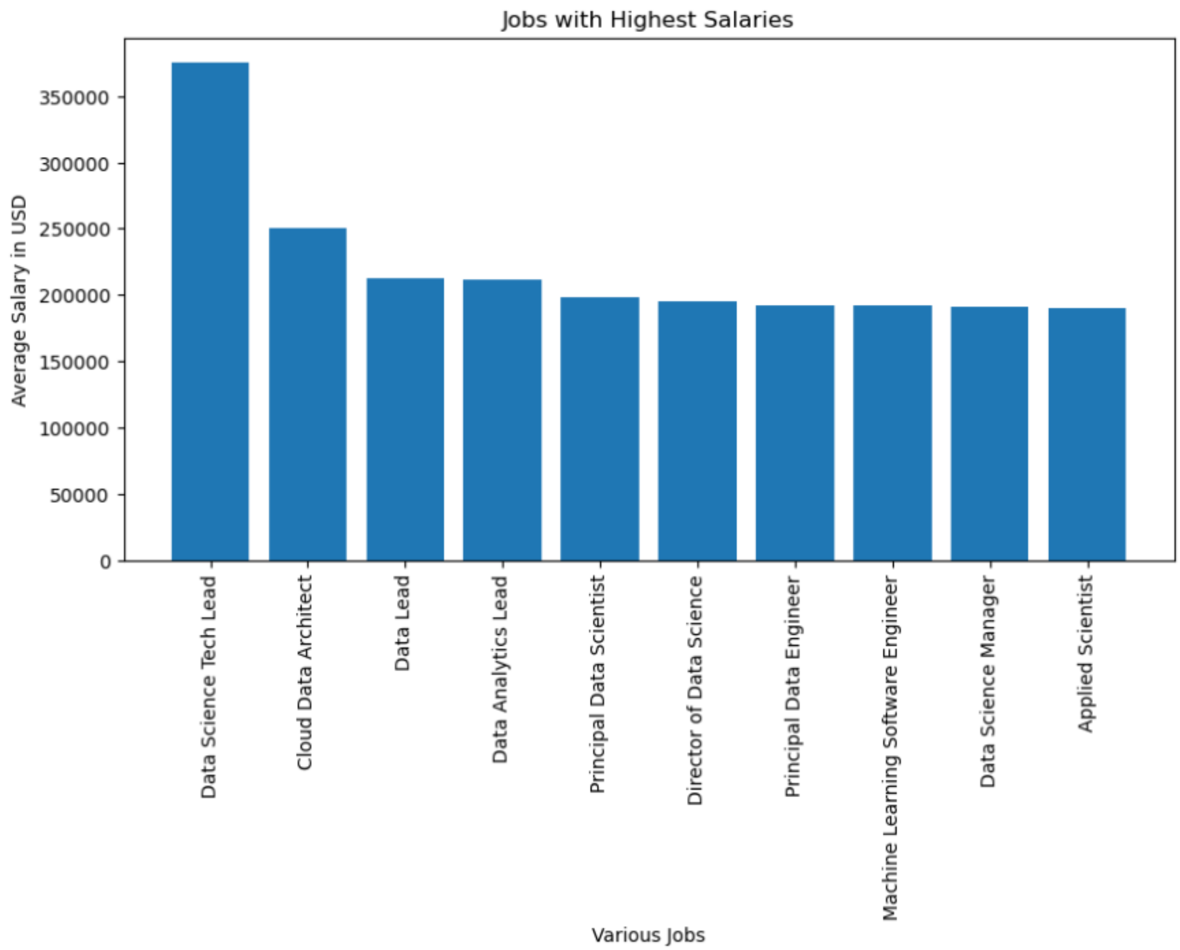

Figure 19 Highest paid job.

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

The code above computes the average salary for each experience level using the construct group by salaries and calculate the average salary or the mean. Then a bar graph is plotted of avg_salary and experienced_on with the help of Matplotlib. avg_salary is the average salary per salary which is on the x-axis and experienced_on are the different levels in which the existing employees operate are on the y-axis. Your graph is then plotted using plt.show(). This code demonstrates how salaries differ from one state of experience to the next.

```
[24]: # Write a python program to find out salaries based on experience level. Illustrate it through bar graph.
experience_level = data_frame.experience_level
salaries = data_frame.salary_in_usd
avg_salary = salaries.groupby(experience_level).mean()

plt.figure(figsize=(10, 6))
plt.bar(avg_salary.index, avg_salary.values)
plt.xlabel('Experience levels')
plt.ylabel('Average Salary based on experience levels')
plt.title('Salaries based on the experience levels')
plt.xticks(rotation=0)
plt.show()
```

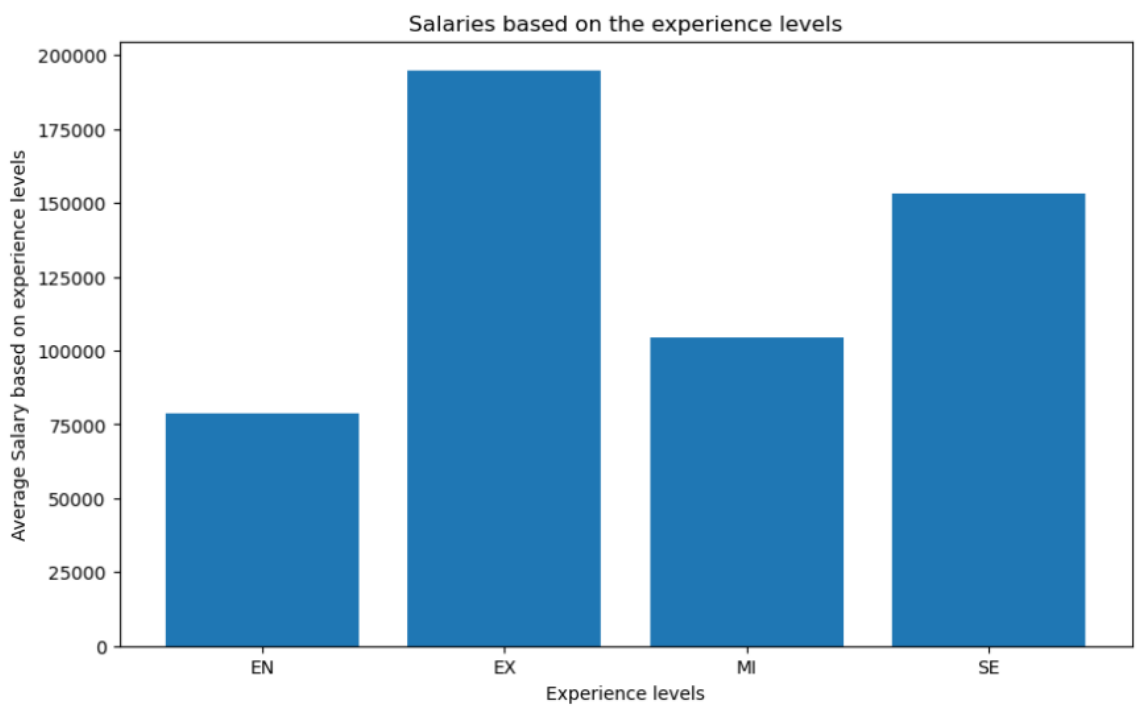

Figure 20 plotting Salaries based on experience level.

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

Below is a Python program that first loads a dataset from a CSV file and then defines a function, plot_histogram_and_boxplot(), to plot the histogram and boxplot for any selected variable. The function takes the column name as input and plots the histogram on the left subplot and boxplot on the right subplot of a 1×2 grid. The user is asked to enter the column name to plot. If the entered column name exists in the DataFrame, the program calls the plot_histogram_and_boxplot() function to generate the plots; otherwise, it will print a message that the column is not found. The histogram shows the frequency distribution of the selected variable, while the boxplot shows the variable's distribution and any outliers.

```
[58]: # Write a Python program to show histogram and box plot of any chosen different variables. Use proper label
data_frame = pd.read_csv("dataset.csv")
# Box plot and histogram plotting functions
def plot_histogram_and_boxplot(column):
    plt.figure(figsize=(10, 5))
# Histogram
    plt.subplot(1, 2, 1)
    plt.hist(data_frame[column], bins=10, color='green', edgecolor='black')
    plt.title(f'Histogram of {column}')
    plt.xlabel(column)
    plt.ylabel('Frequency')
# Box plot
    plt.subplot(1, 2, 2)
    plt.boxplot(data_frame[column])
    plt.title(f'Box plot of {column}')
    plt.ylabel(column)
    plt.tight_layout()
    plt.show()
#Plotting a column using user input
column_to_plot = input("Enter the column name you want to plot: ")
if column_to_plot in data_frame.columns:
    plot_histogram_and_boxplot(column_to_plot)
else:
    print("Column not found in the DataFrame.")

Enter the column name you want to plot:  salary_in_usd
```

Figure 21 Histogram and box plot of any chosen different variables.

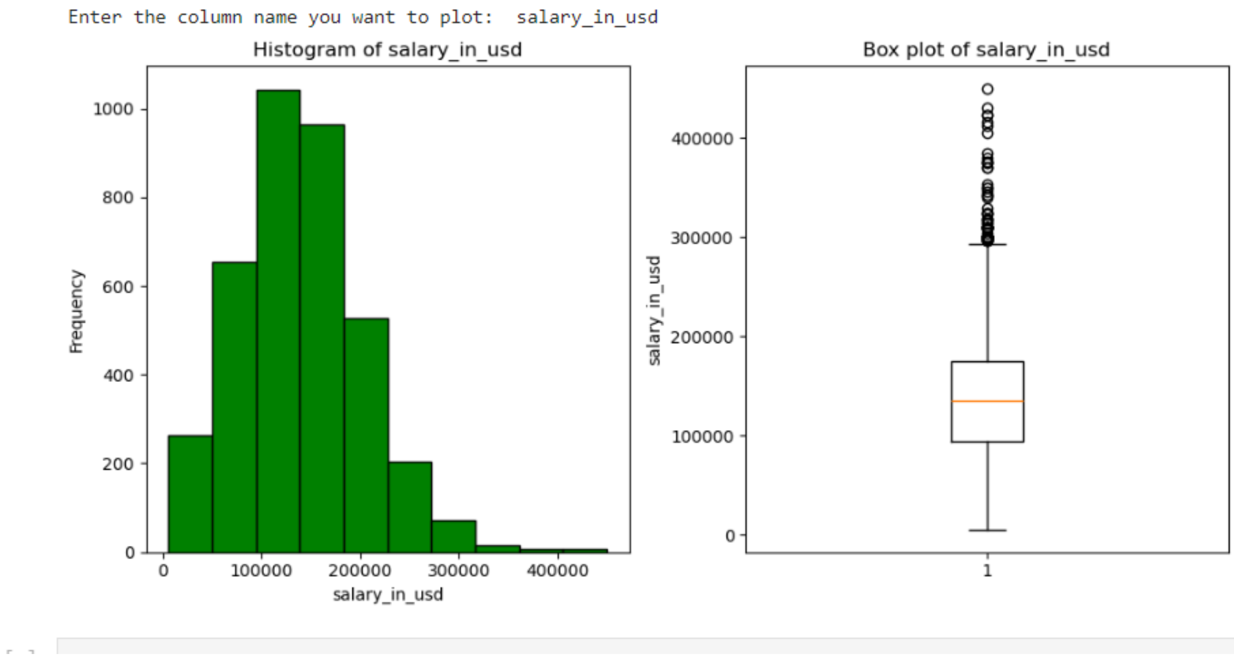

Figure 22 Histogram and Box plott of salary_in_usd.

5. Conclusion

Throughout this project, we have taken a deep dive into the dataset, examining its many elements, and revealing insights into several dimensions. We started with the most basic task of loading the dataset and making sure it was well-suited for analysis. We then spent a significant amount of time preparing the data, restructuring, handling NA values, and ensuring data quality in general. Data analysis followed shortly after with careful inquest to arrive at meaningful conclusions. Whether it was looking over calculations for the summary statistics of the dataset or investigating the strongest correlations between different variables, we were intimately familiar with the data every step of the way. Visualizations proved to be excellent tools for understanding due to the inherent complexity they can encode into easily understood shapes and symbols. From bar graphs on job titles distribution to box plots on the disparity between salaries by experience, each graph was a valuable look into the dataset's narrative.

In addition, we provided users with the opportunity to interact with the data, to observe the data objects they had chosen and to understand how their choice affected the visual representation along with other conditions in real time. We found this dynamic to be highly engaging for users and an enablement for a fully personalized learning experience. Overall, the project provides an example of how Python and various data science tools can be used for unveiling the treasure of essential data hidden in datasets. By combining the power of analytical methods with the benefits of visualization and user-friendly interactivity, we set out for a discovery journey, which has discovered the secrets of the dataset in question and paved the way for new fascinating discoveries and analytical opportunities.

6. References

Numpy tutorial. Available at:

https://www.w3schools.com/python/numpy/

Python data types. Available at:
https://www.w3schools.com/python/python_datatypes.asp

Jaiswal, S. (2019) *Data preparation with Pandas*, *DataCamp*. Available at:
https://www.datacamp.com/tutorial/data-preparation-with-pandas

Robinson, S., Hanna, K.T. and Biscobing, J. (2024) *What is data exploration?*, *Business Analytics*. Available at:
https://www.techtarget.com/searchbusinessanalytics/definition/data-exploration

GeeksforGeeks (2024) *Data Analysis with python*, *GeeksforGeeks*. Available at:
https://www.geeksforgeeks.org/data-analysis-with-python/

Matplotlib histograms. Available at:
https://www.w3schools.com/python/matplotlib_histograms.asp

What is data preparation? processes and example (no date) *Talend*. Available at:
https://www.talend.com/resources/what-is-data-preparation/