**Shai-Assignment**

(Alaa Aldeen Assi)

1-Basic Data Exploration:

import pandas as pd

import numpy as np

# Load your dataset

df = pd.read\_csv('/content/SalariesX.csv')

df.head()

rows = len(df.axes[0])

cols = len(df.axes[1])

print("Number of Rows: ", rows)

print("Number of Columns: ", cols)

print(df.info())

or we can use df.shap and it will return the numbers of rows and columns

and I used df.info to provide us with essential details such as the total number of non-null values, data types of each column and more …

2-Descriptive Statistics

mean\_salary = df['BasePay'].mean()

median\_salary = df['BasePay'].median()

mode\_salary = df['BasePay'].mode()[0]

min\_salary = df['BasePay'].min()

max\_salary = df['BasePay'].max()

std\_dev\_salary = df['BasePay'].std()

print("The mean salary : ",mean\_salary)

print("The median salary : ",median\_salary)

print("The mode salary : ",mode\_salary)

print("The min salary : ",min\_salary)

print("The max salary : ",max\_salary)

print("The std dev salary : ",std\_dev\_salary)

//You asked to make the statistics on the salary column but we don’t have that column in the csv file so I made the statistics on the BasePay column I hope that is fine.

3- Data Cleaning

df[' BasePay '] = df[' BasePay '].fillna(df[' BasePay '].mean())

Imputing missing values with a constant: This method involves replacing missing values with a constant value. The choice of constant depends on the nature of the data and I think most missing data belong to the BasePay and OtherPay columns so the mean might be a good choice.

4- Basic Data Visualization

import matplotlib.pyplot as plt

   df['BasePay'].hist(bins=30, color='skyblue', edgecolor='black')

   plt.xlabel('Salary')

   plt.ylabel('Frequency')

   plt.title('Salary Distribution')

   plt.show()

   df['JobTitle'].value\_counts().plot(kind='pie', autopct='%1.1f%%')

   plt.title('Employee distribution per job title')

   plt.show()

//You asked to to represent the proportion of employees in different departments but in the csv file I didn’t find the department column so I used the JobTitle column.

5-**Grouped Analysis**

avg\_salaries = df.groupby('JobTitle')['BasePay'].mean()

print(avg\_salaries)

 max\_salary\_byJobTitle = average\_salaries.idxmax()

print(max\_salary\_byJobTitle)

max\_salary\_byJobTitle it will contains the name of the jobtitle which has the max salary.

6- **Simple Correlation Analysis**

 correlation = df['BasePay'].corr(df['Year'])

 print(correlation

   plt.scatter(df['Year'], df['BasePay'])

   plt.xlabel('Year')

   plt.ylabel('Salary')

   plt.title('Scatter Plot of Salary vs Years ')

   plt.show()

7- **Summary of Insights**

**We have a csv file contains of the following columns ( 'Id', 'EmployeeName', 'JobTitle', 'BasePay', 'OvertimePay', 'OtherPay', 'Benefits', 'TotalPay', 'TotalPayBenefits', 'Year', 'Notes', 'Agency', )**

**And we made some tasks on this data:**

1. **Grouped Analysis: The data was grouped by the JobTitle column , and summary statistics were calculated for each group. This allowed us to compare the average salaries across different groups and see the difference between the salaries depending on the JobTitle.**
2. **Visualizations**: Histograms, bar charts, and pie charts were used to visualize the distribution of salaries and the count of employees depending on their JobTitle. These visualizations can provide valuable insights into the data, such as identifying outliers in salary distributions or understanding the distribution of employees across different Job Titles.
3. **Scatter Plot**: A scatter plot was plotted to visualize the relationship between salary and year. This can help identify trends in the data, such as whether salary tends to increase with the years.

So we saw how data analysis helped us in understanding the data and summarize the results to help us in collecting information.